Humboldt-Universität zu Berlin
School of Business and Economics
Institute for Statistics and Econometrics
Ladislaus von Bortkiewicz Chair of Statistics

Programming and evaluation of Shiny applications for lectures

Master's Thesis

by

Jonas Gärtner

(564502)

Advisor

Dr. Sigbert Klinke

Examiners

Prof. Dr. Wolfgang Karl Härdle

Prof. Dr. Stefan Lessmann

July 18th, 2017

Acknowledgements

I would like to thank my parents for continuous support and unconditional love.

Also, thank you to Erin:

Who I love for stopping for every dog and every flower,

to say *hi* to one and to smell the other.

Abstract

Statistics education can be enhanced by technology. This thesis describes the theory and development of six **Shiny** web applications. Those applets are designed specifically for students taking introductory classes to statistics or data analysis, lectured by the Ladislaus von Bortkiewicz Chair of Statistics from the Humboldt-Universität zu Berlin. The apps cover among other topics: cluster analysis, regression, and data visualization. The thesis guides through the technical implementation and the iterative improvement of the applets. Evaluating the apps with usability testing played a crucial role in the development. Lastly, future additions and improvements are discussed.

Keywords: R, Shiny, statistics education, data analysis, usability testing

Contents

List of Abbreviations

BIC	Bayesian Information Criterion
CART	Classification And Regression Trees
CP	Complexity Parameter
DBSCAN	Density-based spatial clustering of applications with noise
EM	Expectation-Maximization
GAISE	Guidelines for Assessment and Instruction in Statistics Education
k-NN	k-nearest-neighbor
1R	1-rule

List of Figures

1 Introduction

The following thesis describes six web applications, developed for the Ladislaus von Bortkiewicz Chair of Statistics at Humboldt-Universität zu Berlin. The applets are a tool for students to get engaged with statistical methods. The topics of the methods cover a wide spectrum and range from regression to cluster analysis. The apps are an addition to the already existing apps provided by the Ladislaus von Bortkiewicz Chair (Härdle et al., 2015). The proclaimed goal is to support lectures and excercise classes with the applets, by providing application of methods without the need for students to program.

Teaching statistics has significantly changed in the last twenty years. It was formerly very common to not use any kind of technology. Since the 21st century, most classes have at least one computer projecting on a screen or every student has their own computer (Chance et al., 2007). Fundamental concepts like randomness, sampling, and variability can be hard to understand without any sort of computer simulation. How difficult teaching statistics can be, reveals the existence of journals like *Journal of Statistics Education* and *Technology Innovations in Statistics Education*. In 2005, the Guidelines for Assessment and Instruction in Statistics Education (GAISE) were endorsed by the American Statistical Association. The GAISE suggests to "use technology for developing concepts and analyzing data" (2005). Lane and Tang (2000) found that training with computer simulations lead to better results than training with a textbook. Also, McDaniel and Green (2012) showed the effectiveness of applets in learning statistics. They confirmed the improvement of students' understanding of underlying concepts through the use of applets. Untill the introduction of **Shiny** (Chang et al., 2017) in 2012 the creation of applets needed a lot of effort and it was questionable if the programming was feasible (Härdle et al., 2007). Even the updated GAISE from 2016 acknowledges the mighty **R** (R Core Team, 2016) framework **Shiny**. It postulates **Shiny** to be the "future direction of applets and interactive visualizations" (GAISE, 2016).

The applets programmed in **Shiny** each fulfill several goals. Most of them are about understanding the influence of certain parameter changes for different statistical methods. Students are able to experiment with those methods on several data sets and variables.

This thesis is devided in three main sections. Section 2 holds the theoretical background used in the applets. It provides a short introduction to each topic. Goals and technical implementations of the apps are covered by Section 3. Section 4 contains the evaluation process through usability testing. The thesis ends with a discussion of the work and future improvements. The code of the apps is provided in a CD and the thesis refers to the corresponding

GitHub site.

2 Theory

The topics implemented in the applets are introduced in the following section. Basic theoretical knowledge is provided and explanations only go into detail if neccessary for the apps' understanding.

2.1 Visualization

Graphics can be used for explorative data analysis and presenting results (Chen et al., 2008). Visualizations have the power to communicate complex relations and most difficult information in a small space. They enable the finding of patterns, relationships, and testing for assumptions. Often better than tables, data visualizations can reveal insights into the structure of data (Chambers et al., 1983). Edward Tufte dedicated a whole series of books to principles for good graphical practice (e.g. The Visual Display of Quantitative Information (2001)). The increase in computational power and data complexity lead to a series of new visualization methods in the last 60 years (Friendly, 2006), most of them dealing with multivariate data. This chapter will introduce and show practical applications of graphics that are part of the lectures from the Ladislaus von Bortkiewicz Chair of Statistics. All of the visualization methods are implemented in at least one of the programmed **Shiny** applets.

2.1.1 Andrews curves

Andrews curves are a visualization technique for multivariate data and were introduced by Andrews in 1972. The curves can help to find structure in a data set. Each observation $x = x_1, x_2, x_3, .., x_d$ defines a finite Fourier series. The **R** package **andrews** (Myslivec, 2012) offers four different type of curves:

1. $f(t) = x_1/(\sqrt{2}) + x_2 * sin(t) + x_3 * cos(t) + x_4 * sin(2t) + x_5 * cos(2t) + ...$
2. $f(t) = x_1 * sin(t) + x_2 * cos(t) + x_3 * sin(2t) + x_4 * cos(2t) + ...$
3. $f(t) = x_1 * cos(t) + x_2 * cos(\sqrt{2t}) + x_3 * cos(\sqrt{3t}) + ...$
4. $f(t) = 1/(\sqrt{2}) * (x_1 + x_2 * (sin(t) + cos(t)) + x_3 * (sin(t) - cos(t)) + x_4 * (sin(2t) + cos(2t)) + x_5 * (sin(2t) - cos(2t)) + ...)$

The function plots one line for every observation x between -π and π. Plotting all ob-

servations of a data set reveals subgroups, outliers and a general structure. Andrews curves have some practical properties. For example, the function of the mean observation is the same as the pointwise mean of the functions of the observations,

$$f_{\bar{x}} = \frac{1}{N} \sum_{i=1}^{N} f_{xi}(t).$$

Also, the distance between two functions is proportional to the Euclidean distance of those data points (Garca-Osorio and Fyfe, 2005). The order of the variables has a strong impact on the shape of the curve. Some orders might display less insights than others. It is recommended to try out multiple arrangements of the variables (Embrechts and Herzberg, 1991) or to use the order proposed by the Principal Component Analysis (Andrews, 1972, 135). Andrews curves also tend to overplotting when data points are fairly similar or when there are too many observations and, thus, a differentiation of the curves is not possible.

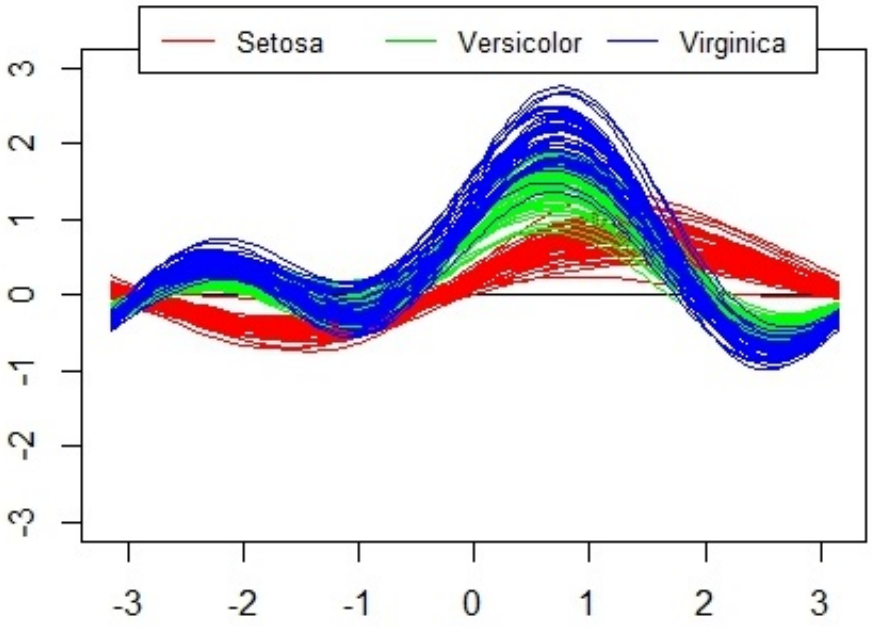


Figure 1: Andrews curves of iris data set

Figure 1 shows a use case for Andrews curves. Instead of looking at a table with 150 observations, each containing four variables, the general structure of the data set is revealed by Andrews Curves. It can be immediately observed that *Setosa* iris flowers are relatively different from the other two. *Versicolor* and *Virginica* have slightly different curves, but cannot be differentiated as well. Figure 1 is also an example for overplotting, which comes with too many observations. Many of the *Versicolor* curves are hidden.

2.1.2 Parallel coordinates

Parallel coordinates are another way of visualizing multivariate data (Inselberg, 1997). Each variable in a data set belongs to a vertical axis and every observation is a connecting line, going from axis to axis. Similar to Andrews curves, Parallel coordinates reveal a general structure of a data set and clusters of similar observations can be found. Also, the order of the variables has an impact on the pattern and should be varied. One advantage over Andrews curves is that Parallel coordinates not only reveal the structure of the data, but also show the relationship of variables to each other. Crossing lines between two axes can mean negative correlation between those variables and parallel lines can be a sign for positive correlation (Inselberg, 2009). Overplotting is again a problem, but can be solved by clustering of the data (Wegman and Luo, 1997). The interpretation of Parallel coordinates is easier when the variables are on the same scale, which is, therefore, the default method.

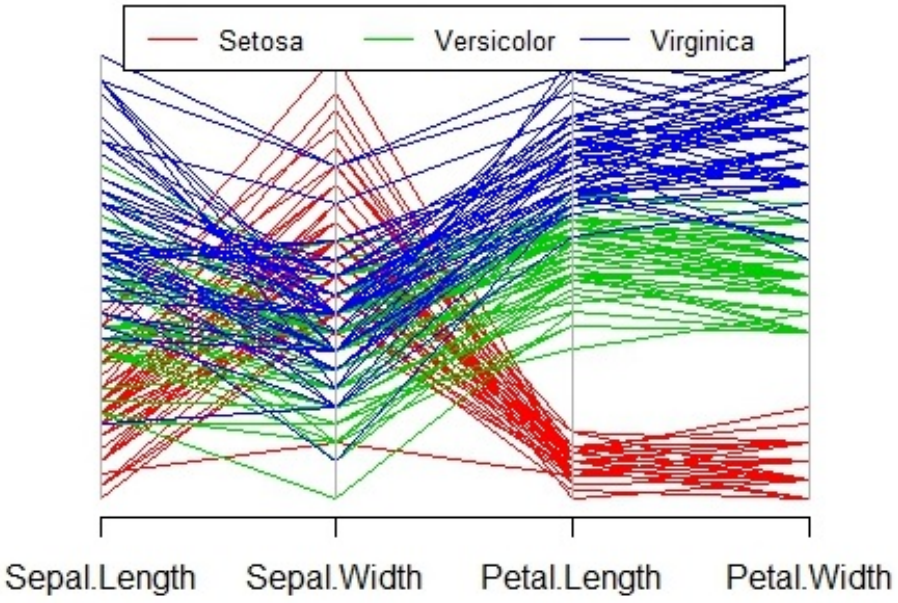


Figure 2: Parallel coordinates of iris data set

Figure 2 shows a use case for Parallel coordinates. Similar to Andrews curves, the general structure of the data is uncovered. The species *Setosa* is easy to differentiate from *Versicolor* and *Virginica*. Additionally, the graphic shows how observations score on different variables. Also, correlations can be observed. The plot implies that *Sepal.Width* is negatively correlated with *Sepal.Length* and *Petal.Length*, considering the crossing of the lines.

2.1.3 Star plot

Similar to Parallel coordinates is the Star plot, otherwise referred to as Spider or Radar chart. The Star plot can be interpreted as a Parallel coordinates plot with polar coordinates. About 20 variables can be displayed (Lanzenberger, 2003). Every variable is one ray and each observation creates a star-formed figure. The higher the value is for an observation on a variable, the longer the ray of the star (Friendly, 1991). The Star plots can be used to get a quick overview of the data. Outliers can be recognized quickly because the shape of the star looks different from the rest. Also, groups with similar values can be identified. Star plots are limited to a view hundred observations, since more stars become hard to process (Hinum, 2006).

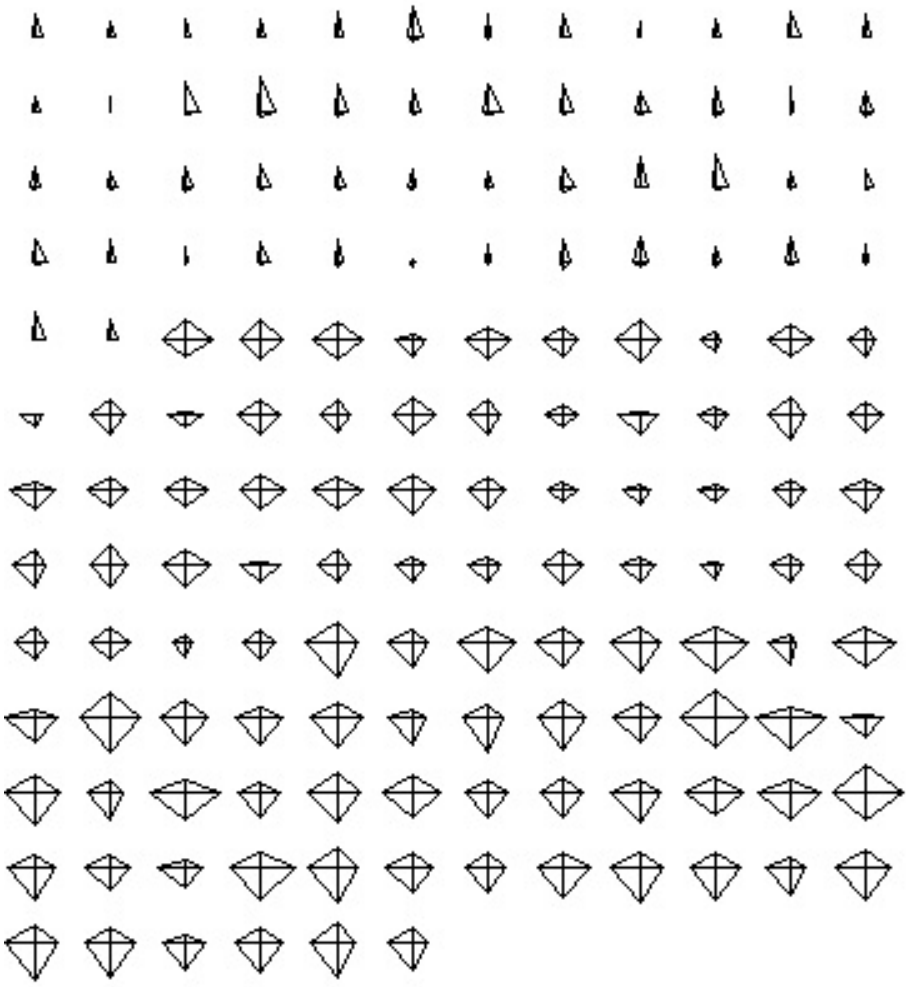


Figure 3: Star plots of iris data set

Each of the 150 observations from the iris data set is displayed as a star in Figure 3. Each star has four rays which represent the four variables: *Sepal.Length*, *Sepal.Width*, *Petal.Length*,

and *Petal.Width*. At first glance the three types of flowers can be recognized by the different star shapes. In this case, the data was already ordered and, therefore, the plot is easy to interpret. Unordered data and a higher number of observations would expose the limitations of a Star plot.

2.1.4 Chernoff faces

Another way to display multivariate data is the use of Chernoff faces (Chernoff, 1973). Each feature of the face represents a different variable and each observation is assigned to one face. Chernoff states that "people grow up studying and reacting to faces all of the time" (1973, 363) and that even small differences in faces can be recognized. This advantage in human processing should be used in analyzing data. The effect of different values in a variable depends on the corresponding feature. A relative high value might result in a wider smile or change the angle of the eyebrows. Therefore, the order of variables has a great impact on the appearance of the faces.

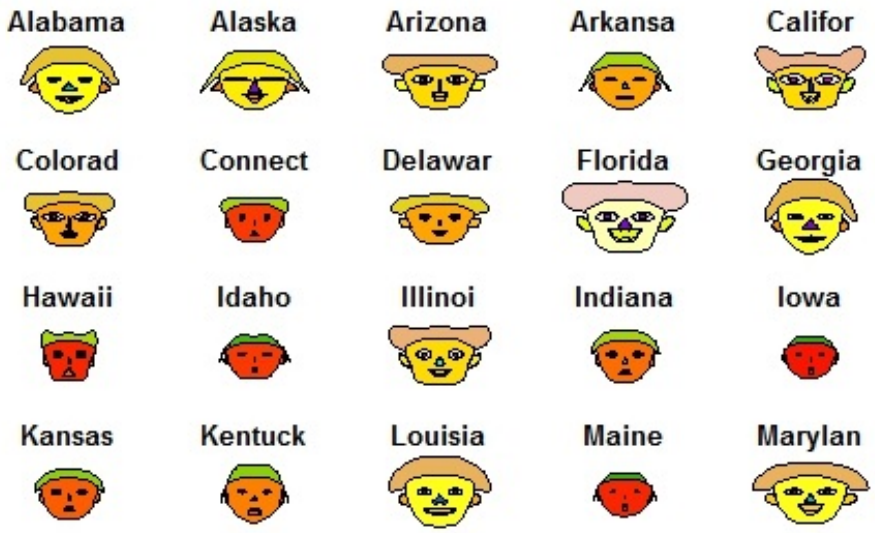


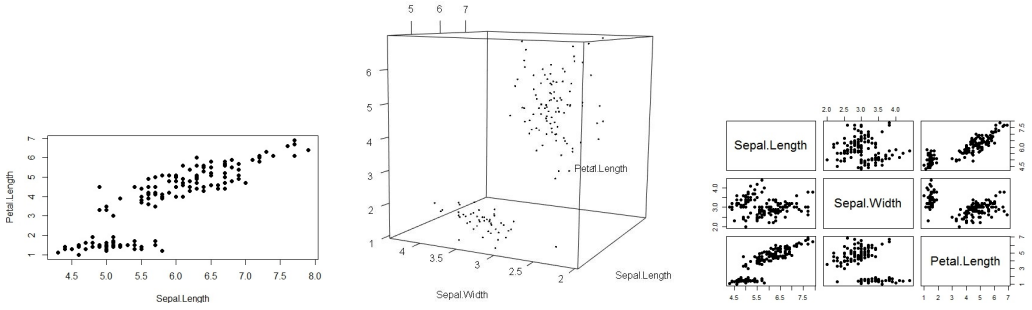
Figure 4: Chernoff faces of USArrest data set

The Chernoff faces in Figure 4 display crime data of 20 different states in the US. Without knowing any more details, states with similar faces can be grouped together. For example, Alabama and Georgia seem to have similar characteristics, so do Iowa and Maine, where crime statistics are relatively low. Figure 4 was created with the **aplpack** package (Wolf, 2014). The function `faces()` can handle up to 15 different variables without the use of color. If not all variables are needed, variables might correspond to more than one feature in the

face. Chernoff faces are most useful for the initial analysis of the data and can be helpful for cluster and discrimination analysis (Chernoff, 1973, 363).

2.1.5 Scatter plot

A widely-used tool in data visualization is the scatter plot. Points are plotted on cartesian axes, where each axis is representing one variable (Gentle, 2002). The standard scatter plot is two-dimensional and shows the relationship of two continuous variables. The three-dimensional scatter plot and scatter plot matrix are alternative methods for complex data sets with more than two variables of interest. Scatter plots are useful to reveal outliers, clusters, and relationships (Chambers et al., 1983, 86-87).



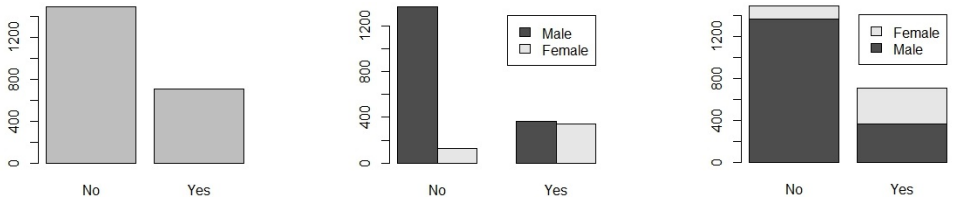
(a) 2d scatter plot (b) 3d scatter plot (c) scatter plot matrix

Figure 5: Scatter plots of iris data set

The two-dimensional scatter plot in Figure 5a reveals the existence of two clusters and a positive correlation between the variables *Sepal.Length* and *Petal.Length*. The scatter plot matrix in Figure 5c shows three pairs of relationships. Three-dimensional scatter plots can be hard to read, but useful if they can be rotated (Figure 5b).

2.1.6 Bar chart

Absolute and relative frequencies for categorical variables can be displayed by bar charts. Every category is displayed by one bar and the height is proportional to the frequency (Schlittgen, 2004).Thus, categories can be compared very quickly. To plot two categorical variables at a time, bar charts can be stacked or grouped. Stacked bar charts help to compare the total value, but make it harder to compare different categories between each other. Grouped bar charts help to compare different categories, but make it difficult to differentiate the total value.



(a) bar chart (b) grouped bar chart (c) stacked bar chart

Figure 6: Bar charts of Titanic survival data

The bar chart in Figure 6a compares the number of survivors to non-survivors after the sinking of the Titanic. In the grouped bar chart 6b the same observation is made, but for the different genders. It can be seen that slightly more men than women survived. The stacked bar chart 6c is best to compare the total number of survivors and non-survivors.

2.1.7 Spine plot

In a spine plot, different to a bar chart, all of the bars have the same height. The width varies proportional to the frequency in the category. This makes it much easier to compare the proportion of the highlighted areas between the different categories (Theus, 1997). The bars differ in their width in the same proportion, as the corresponding stacked bar chart differs in height.

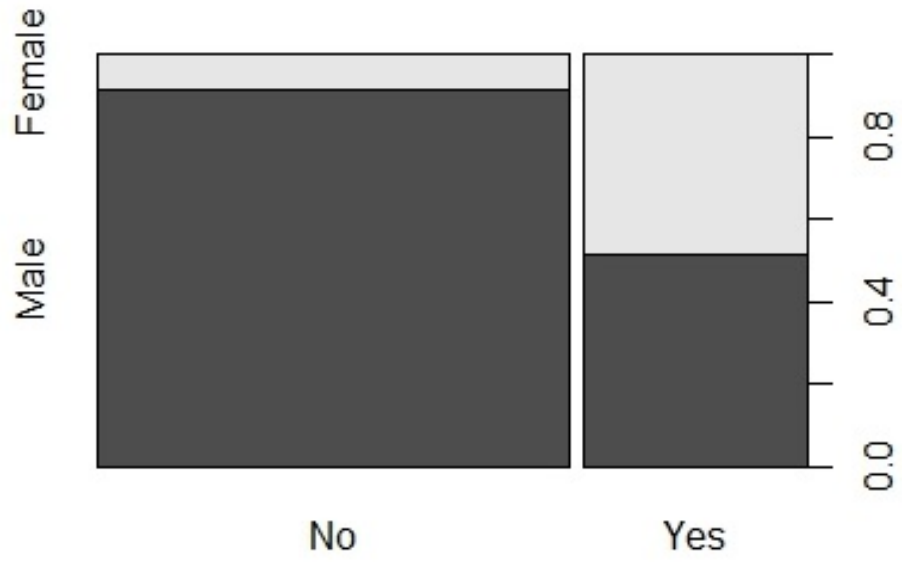


Figure 7: Spine plot of Titanic survival data

The spine plot in Figure 7 makes it easier to compare the relative amounts of women in each group. Gender did seem to have an influence on the chance of surviving, since the

relation of man and women differs between survivors and non-survivors.

2.1.8 Mosaic plot

The extension of a spine plot is the mosaic plot, where even more categorical variables can be used. Mosaic plots display contingency tables by assigning each cell to a tile with the size of the proportional cell count. This method can be generalized to multi-way tables (Friendly, 1994). If there are more than two variables, subsequent horizontal and vertical recurring splits are used. Each new variable is conditional to the last added variable (Meyer et al., 2003). Output and possible information gain are therefore greatly dependent on the order of the variables. Mosaic plots are mostly used to explore independence between the categorical variables. If there is independence, the boxes allign in a grid, since all proportions are equal throughout the categories.

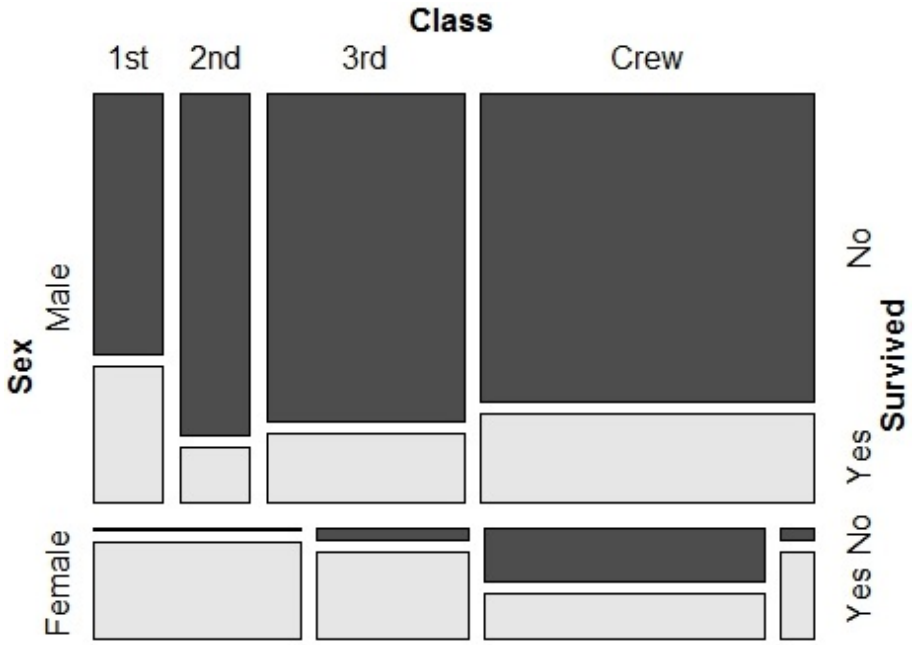


Figure 8: Mosaic plot of Titanic survival data

The outcome variable the researcher is most interested in should be the last variable added to a mosaic plot. In Figure 8, it is the variable *Survived*. Since the boxes do not line up in a grid, dependence of the variables can be assumed.

2.1.9 Parallel coordinates for categorical data

The original idea to generalize Parallel coordinates to categorical data was introduced by Schonlau (2003) as the Hammock display. Parallel-sets (Kosara et al., 2006) and the Common-

angle plot (Hofmann and Vendettuoli, 2013) are further developments to ease the interpretation of the line width. The layout is similar to Parallel coordinates, where each variable belongs to a vertical axis. Instead of individual observations, there is a frequency based display. One advantage over mosaic plots is that variables are represented independently from each other. This makes them easier to understand when the amounts of variables increase. The relationship between neighboring variables can be analyzed. Again, the order should be varied to gain more information about the data.

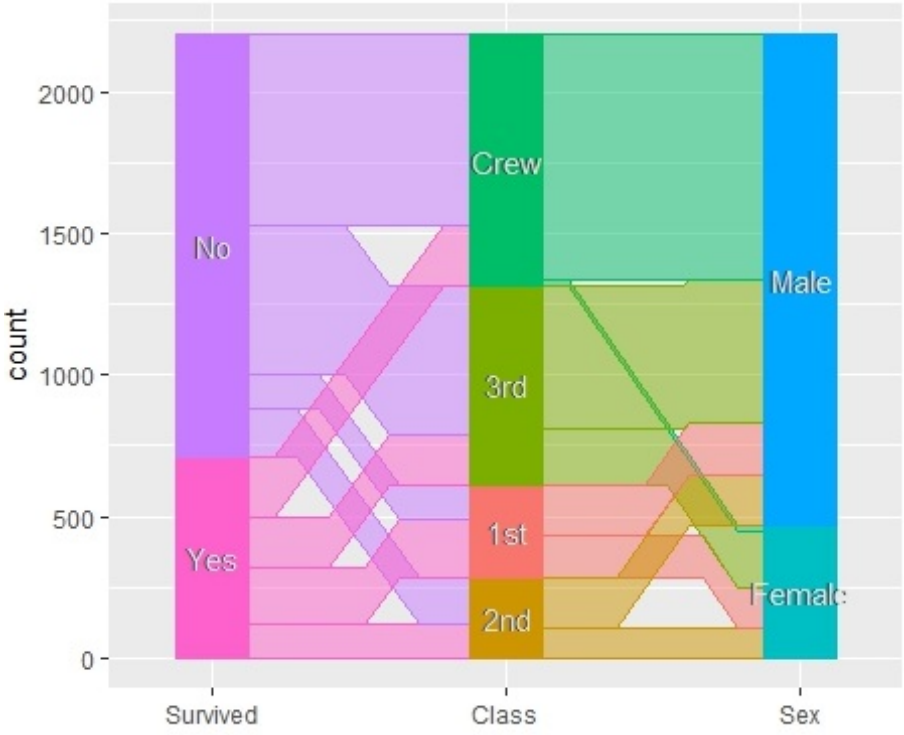


Figure 9: Common-angle plot of the Titanic survival data

In Figure 9 the relative relationship between the variables *Class/Survived* and *Class/Sex* is displayed. It can be seen that the likelihood of surviving in the *1st Class* was much higher than for a person in the *Crew*. Also, almost no *Female* passengers belonged to the *Class Crew*. Parallel coordinates hold less information than mosaic plots, but can be easier to understand and answer specific questions.

2.2 Association statistics

Measures of association can quantify the relationship of variables (Liebetrau, 1983). Finding an association might not be enough; the direction and strength is important. Usually, measures of association are scaled between 0 and 1, where 0 means no relationship and 1

shows a perfect relationship. If they range from -1 to 1, also the direction of association is determined (Gingrich, 2004). Besides coefficients, hypothesis testing is also possible. Traditionally the H_0 states no relationship between the variables. The following paragraphs will focus on statistics between two nominal (also referred to as categorical) variables.

2.2.1 Chi-square test

A relationship between to categorical variables, can be found by the *chi-square* test for independence by Pearson (1900). The test can (among other things) evaluate if two variables are independent of one another. For a *chi-square* test, the data must be in a contingency table. The *chi-square* statistic compares the expected frequency under independence with the observed frequency. The statistic is defined as

$$\chi^2 = \sum_{i=1}^{I} \sum_{j=1}^{J} \frac{(O_{ij} - E_{ij})^2}{E_{ij}},$$

where O_{ij} is the observed frequency in category ij and E_{ij} is the expected number of cases in category ij under independence (Gingrich, 2004, 706). The expected values are subtracted from the observed ones, squared, and divided by the expected. Those results are summed up and lead to the test statistic χ^2. If there is a small difference between O_{ij} and E_{ij} for each category, the null hypothesis is true and the variables are independent. χ^2 is large if there is a big difference between O_{ij} and E_{ij} in each category. The *chi-square* statistic for independence is approximated by the *chi-square* distribution. Problems occur with small expected frequencies. The expected frequency should not be smaller than 5 (Cochran, 1952). If such a case occurs in a 2x2 table, the Yates correction or the Fisher exact test can be used. The Fisher exact test can also be extended to larger tables (Howell, 2011).

An alternative to the Pearson *chi-square* statistic is the likelihood ratio *chi-square*. It is composed on the likelihood of the data under the H_0 relative to the maximum likelihood,

$$\chi^2 \approx -2 \sum_{i=1}^{I} \sum_{j=1}^{J} O_{ij} log\big(\frac{O_{ij}}{E_{ij}}\big).$$

With increasing sample sizes the two *chi-square* statistics converge (Howell, 2011, 2).

2.2.2 Measures of association based on chi-square

The *chi-square* statistics test if there is a relationship between two variables, but there are a series of problems, which the measures of association try to solve. A test can tell if there is a relationship, but does not explain the characteristics. How a change in one variable affects

the other variable is not determined by a *chi-square* test. Also, *chi-square* values are hard to compare. Only with the same sample size and amounts of cells, can two relationships be compared by *chi-square* (Gingrich, 2004, 769-770). The amounts of cells are represented by the number of rows and columns in a table, these are called dimensions of a table. The following measures of associations are created to handle different sample sizes and dimensions.

Phi takes the sample size into account to get a better idea about the effect size. The coefficient is traditionally introduced to be used for a 2x2 table and is defined as

$$\Phi = \sqrt{\frac{\chi^2}{n}},$$

with n being the sample size. *Phi* is 0 if *chi-square* is 0, i.e. there is no relationship. The maximum value of *Phi* is generally 1 and is interpreted as a perfect relationship. The *Phi* coefficient creates the same result as the Pearson correlation for two binary variables and therefore the same guidelines for effect sizes apply. Every value of 0.5 and higher means a large effect, 0.3 a medium and 0.1 a small effect (Cohen, 1988).

Cramer's V can be used for tables with all dimensions and is equivalent to the *Phi* coefficient, if there is a 2x2 table. It is defined as

$$V = \sqrt{\frac{\chi^2}{n(m-1)}},$$

where n is the sample size and m is the minimum value of rows and columns. This measure is a way to compare tables with different dimensions. If there is no relationship, V equals 0 and the stronger the relationship, the larger the value. The maximum is again usually 1 and means a very strong association (Gingrich, 2004, 782-784). The interpretation of small and large effect sizes depends on the degrees of freedom (Cohen, 1988, 222).

The contingency coefficient is also based on *chi-square* and is adjusting for the sample size. The measure of association is defined as

$$C = \sqrt{\frac{\chi^2}{n+\chi^2}}.$$

Phi and the contingency coefficient are closely related. No relationship is represented by $C = 0$. C cannot be greater than 1. It may be even less than 1, when there is a perfect association. This makes the coefficient harder to interpret and *Phi* and Cramer's V are generally preferred (Gingrich, 2004, 778).

2.3 Cluster analysis

Cluster analysis is a class of methods for data reduction and is mostly used for explorative analysis (Schlittgen, 2009). The goal is to summarize observations in homogeneous groups

(clusters). These clusters are created in a way that elements in a group are similar and different to elements in other groups. Two elements are similar if their dissimilarity is small or their proximity is large. The different choices for measurements of dissimilarity and similarity can have a great impact on the result (Everitt et al., 2011). The most commonly used distance between observations, when all variables are continuous, is the Euclidean distance

$$d_{ij} = \Big[\sum_{k=1}^{p}(x_{ik} - x_{jk})^2\Big]^{1/2},$$

where x_{ik} and x_{jk} are the values of the k-th variable for the individuals i and j. The easiest interpretation is the physical distance between two points in Euclidean space. An alternative is the Manhattan distance, which measures the distance by only moving horizontally and vertically, like on a street map.

Since all variables in a data set are seldom measured in the same unit, standardization such as z-scoring is recommended (Mohamad and Usman, 2013). Cluster analysis without standardization can lead to vastly different cluster solutions. After a cluster analysis, the meaningfulness of the cluster solution must be evaluated. No clusters can also be a valid result. To evaluate cluster analysis, silhouette plots can aid with the interpretation and validation (Rousseeuw, 1987). Each silhouette represents a cluster and shows which observations belong with more certainty in their cluster and observations that are closer to other clusters. The average silhouette width evaluates the validity of the cluster analysis.

The classical clustering methods are hierarchical and partition-based. Newer approaches include model-based and density-based clustering. In some cases, graphical solutions like histograms and scatter plots might already be sufficient for clustering (Everitt et al., 2011, 33-60).

2.3.1 Hierarchical clustering

Hierarchical clustering can be divided in agglomerative and divisive methods. The top-down approach, where all elements start in one cluster and then gradually become separated, is called divisive. At the end of that process, each cluster contains only one observation. The bottom-up method, where all elements start in their own cluster and then the clusters gradually join each other, is called agglomerative. In both processes the result is a hierarchy of cluster solutions, which is best presented in a dendrogram (Schlittgen, 2009, 397). The divisive method is much less common, since it can be computationally demanding if all group splits are considered at every stage. The divisive diana-algorithm (Kaufman and Rousseeuw,

1990) looks at every stage only at the cluster with the highest dissimilarity between two observations. To divide this cluster, the observation with the highest distance on average to all other group members initiates the split. Then elements in the old group join the single observation if it is closer. The process runs till all clusters only obtain one element.

The more important hierarchical method is agglomerative clustering. At the start of the process, all clusters only contain one element. In the next step the two groups with the minimal distance will be joined together. It is crucial to mention that not only the distances between elements must be defined (Euclidean, Manhatten etc.) but also those between clusters. One popular method is single-linkage where the distance between two groups (A & B) is defined as the shortest existing distance between two different group members (a & b),

$$d(A, B) := \min_{a \in A, b \in B} d(a, b).$$

This can create long thin clusters, since not all observations in a cluster are considered during the process.

Another popular solution is complete-linkage. Here the distance is defined as the maximum of all distances between the members of the group,

$$d(A, B) := \max_{a \in A, b \in B} d(a, b).$$

This leads to the construction of rather small groups. There is a whole series of alternative agglomerative methods, like average-linkage, median-linkage or the Ward method. Which method should be chosen depends mostly on the data and preferred clusters(Schlittgen, 2009, 399-401).

2.3.2 Partitional clustering

The goal of partitioning is to find a clustering solution with a given number of clusters. This is different to hierarchical clustering, where a number of clusters does not have to be defined in the beginning. To use partitional clustering in an explorative way, one can run a method multiple times with different amounts of clusters. In practice, agglomerative hierarchical clustering is done first to get a better idea about a fitting number of clusters and then followed by partition clustering. The general concept of partitioning is to start with a partition and then in an iterative process move elements between groups to improve the solution to an optimum. The different methods for partition result from different criteria of optimization. The most used partitional clustering method is K-Means, with the Lloyd

Algorithm as the traditional algorithm. First, a random partition with K number of clusters is created. For each cluster the centroid is then calculated. Each element changes to the group it has the lowest distance to the mean. If at least one element changes the cluster, the centroids will be recalculated and distances to the group means are compared again. The criterion function for optimization is the sum of squared deviations from the cluster means μ (Lloyd, 1982),

$$\sum_{i=1}^{k} \sum_{x_j \in S_i} (x_j - \mu_i)^2.$$

K-Means only finds a local optimum and it is therefore recommended to calculate the method multiple times with different starting partitions (Rubin, 1967). If the result stays the same after multiple runs, the solution is trustworthy. Variations of the Lloyd Algorithm are MacQueen's Algorithm and the algorithm by Hartigan and Wong. The MacQueen Algorithm is supposed to be more efficient, since it updates centroids more often (Morissette and Chartier, 2013). Hartigan and Wong compare not only distances in their algorithm, but also the cost of reassigning a point to a different cluster (Slonim et al., 2013).

To overcome the problem of different results depending on the starting partition, K-Means++ was introduced. It again chooses centers at random, but weighs the elements according to their "squared distance squared from the closest center already chosen" (Arthur and Vassilvitskii, 2007). Arthur and Vassilvitskii (2007) also claim that K-Means++ in general is faster and more accurate than K-Means. An alternative to K-Means clustering are K-Medoids methods. The medoid is the most central data point in a cluster. Each element will be assigned to the cluster containing the medoid to which it is nearest. After the rearranging to new clusters, the medoids can change and elements may switch groups again. The optimization is again an iterative process. K-Medoids methods are preferred if the data has more extreme values, since K-Medoids are more robust than K-Means methods (Schlittgen, 2009, 412-413). One generalized version of the K-Means clustering algorithm is Fuzzy C-means clustering. Each element is partially classified into all clusters. It has the advantage that points between two clusters do not have to be forced into a specific cluster. Memberships of the elements lie between 0 and 1 and can be interpreted as probabilities. The information about the second-best cluster is lost in traditional methods (Everitt et al., 2011, 242-245).

2.3.3 Other clustering methods

A different approach to cluster building can be model-based, where it is assumed that the data was created by a model (Manning et al., 2008). The classic approach of maximum

likelihood is used to estimate the model parameters. Iterative Expectation-Maximization (EM) methods for maximum-likelihood estimation (Fraley et al., 2012) are very common. These methods contain a so-called E-step, where conditional probabilities that observations belong to clusters are estimated. The M-step solves the likelihood equation. An appropriate model will be chosen with help by the Bayesian Information Criterion (BIC). No number of clusters has to be defined a priori.

One award winning algorithm is DBSCAN (Density-based spatial clustering of applications with noise). The algorithm is specialized in spatial data and the locating of clusters with arbitrary shape (e.g. spherical, lengthy, or ring-shaped). Only the distance parameter epsilon and the minimal amount of points in a cluster have to be chosen to get an efficient result for large data sets. As opposed to K-Means, the number of clusters does not have to be selected (Ester et al., 1996).

2.4 Regression

Regression analysis estimates the relationship between variables. One or more independent variables are predictors and provide information about the dependent variable. The dependent variable is denoted with Y and the independent variables with X. Usually, the model is directed to describe how $E[Y]$ of the dependent variable changes with different values for the independent variables. All models also contain parameters, which control the behavior of the model (Rawlings et al., 1998). How the parameters are connected to the independent variables depends on the kind of model. There is great variety of models and only a select few will be introduced in this chapter.

2.4.1 Simple linear regression

The most simple linear model contains only one dependent variable. When the independent variable increases or decreases, the true mean of the dependent variable changes by a constant rate:

$$E[Y_i] = \beta_0 + \beta_1 X_i,$$

where $E[Y_i]$ is the true mean, β_1 is the slope, and β_0 the intercept of the line. One assumption is that the observations of the dependent variable Y_i are "random observations from populations of random variables with the mean of each population given by $E[Y_i]$" (Rawlings et al., 1998, 2). To take account for the deviation between Y_i and $E[Y_i]$ a random error term

ϵ_i is added,

$$E[Y_i] = \beta_0 + \beta_1 X_i + \epsilon_i.$$

Further assumptions are that the observations X_i are measured without error. Also, ϵ_i are normally and independently distributed, with mean 0 and the variance σ^2,

$$\epsilon_i \sim NID(0, \sigma^2).$$

In the simple linear model, the parameters β_0 and β_1 have to be estimated from the data since there is a random error in Y_i. The estimates are denoted as $\hat{\beta}_0$ and $\hat{\beta}_1$ and the procedure is called least squares estimation. The estimated mean of Y be $\hat{Y}_i$ for each X_i,

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_1.$$

The estimated parameters $\hat{\beta}_0$ and $\hat{\beta}_0$ are chosen in a way that minimize the sum of squares of the residuals,

$$\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 = \sum e_i^2.$$

Sometimes theory suggests that the dependent variable is expected to be 0 when the independent variable is 0. Then a linear model without an intercept is appropriate,

$$Y_i = \beta_1 X_i + \epsilon_i.$$

To measure how much variance is explained by the independent variable the coefficient of determination R^2 can be used. R^2 ranges from 0 to 1 and is the product moment correlation between Y_i and $\hat{Y}_i$ (Rawlings et al., 1998, 3-10). The coefficient can be used to compare different models between each other. R^2 only finds linear relationships, even if the true relationship might be different. Since R^2 tends to overestimate the effect in the population, the adjusted R^2 is a good alternative (Bortz and Schuster, 2010). The formula is:

$$R^2_{adj} = 1 - \Big[\frac{(1 - R^2)(n-1)}{n-k-1}\Big],$$

where n is the number of observations and k is the number of independent variables.

Also from importance in simple linear regression is the question of whether the regression coefficients are significant. In other words, the hypotheses are

$$H_0 : \beta 1 = 0,$$

$$H_1 : \beta 1 \neq 0.$$

The test statistic t is

$$t = \frac{\beta_1}{s\hat{e}.(\beta_1)},$$

where $s\hat{e}.$ is the estimated standard error (Rawlings et al., 1998, 16-17).

2.4.2 Multiple linear regression

An extension of the simple linear model is the multiple linear regression model. The dependent variable is explained by more than one independent variable,

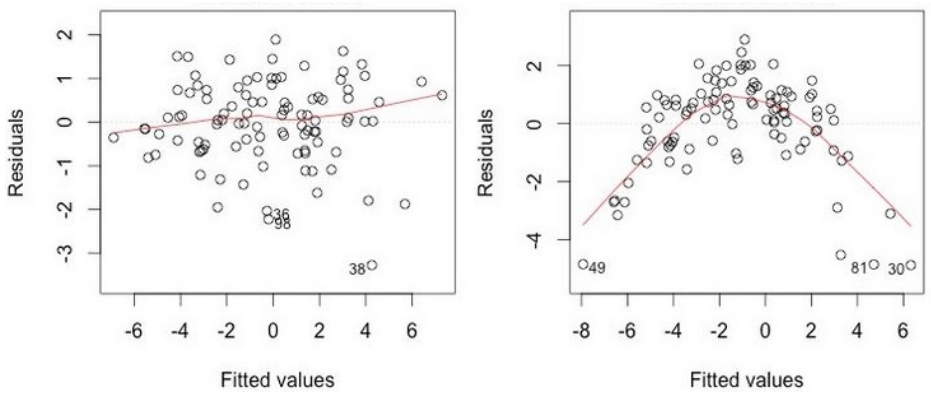
$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \beta_3 X_{i3} + ... + \beta_p X_{ip} + \epsilon_i.$$

The assumptions from the simple linear regression still stand. In practice, the multiple linear regression is much more common, since often the influence from more than one variable is of interest. The multiple linear regression can also hold interaction terms or polynomials. These additions make the linear regression much more flexible and allow for more in-depth analysis.

2.4.3 Residual analysis

After a regression analysis, the validity of the model should be analyzed. Whether a model is correct depends heavily on the correctness of the assumptions made a priori. Three major problem areas are the normality, common variance, and independence of the error terms. The failure of different assumptions have varying consequences. For example, nonnormality of the errors still produces the best linear unbiased estimates, but the test for significance and construction of confidence interval estimates for the parameters will be off. A series of test and graphical diagnosis can reveal false assumptions (Rawlings et al., 1998, 325-335). The rest of this chapter will focus on the diagnosis with graphics, since that will be the focus of the corresponding applet.

One of the most useful graphics is the plot of the residuals against the fitted values. If all assumptions are satisfied, random scattered points should be around $e = 0$ with most of the data being between $e = \pm 2$. Any pattern of the residuals indicates a violation against the assumptions.

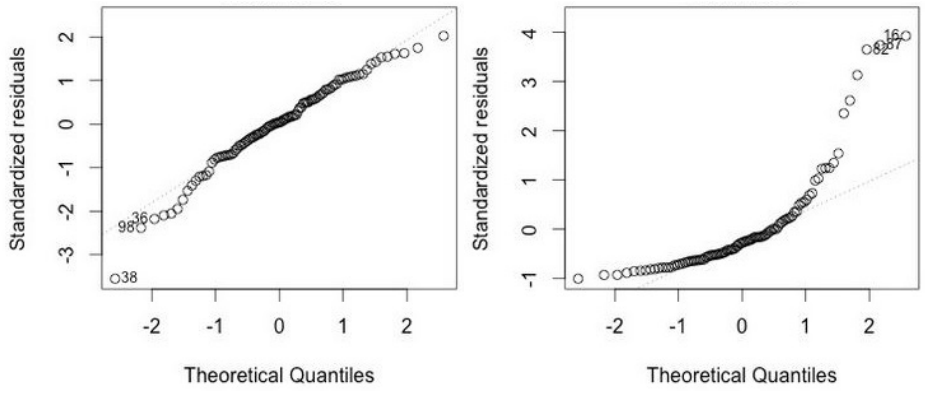


(a) no pattern (b) pattern

Figure 10: Residuals vs. Fitted values plots. Reprinted from Bommae (2015).

The curved shape of the residuals in Figure 10b propose a non-linear relationship which is not explained by the model (Bommae, 2015). The assumption of common variance (homoscedasticity) is violated in Figure 10b.

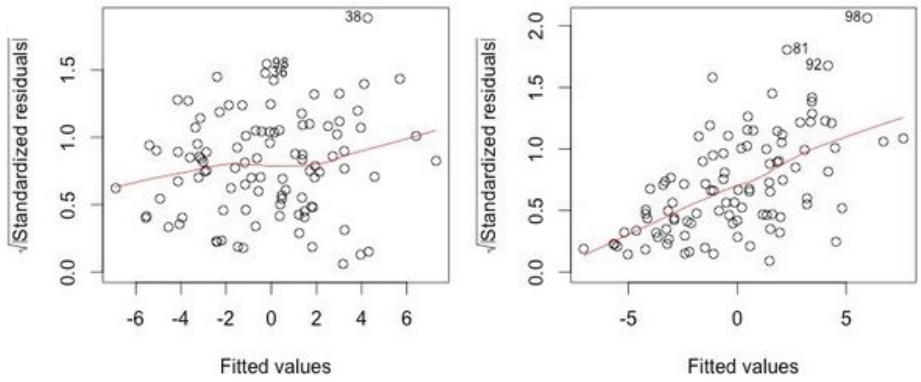
A good option to see if residuals are normally distributed is the Normal Q-Q plot. Theoretical quantiles are plotted against the standardized residuals. If the points do not follow the dotted line, normality of the residuals is not given. Depending on the shape of the points, a variety of attributes can be seen, e.g. heavy-tails, light-tails, or skewedness.



(a) Example 1 (b) Example 2

Figure 11: Normal Q-Q plots. Reprinted from Bommae (2015).

Figure 11a suggests a more normal distribution than Figure 11b. The Scale-Location plot can help to check the assumption of constant variance i.e. homoscedasticity. The fitted values are plotted against the root of the standardized residuals.

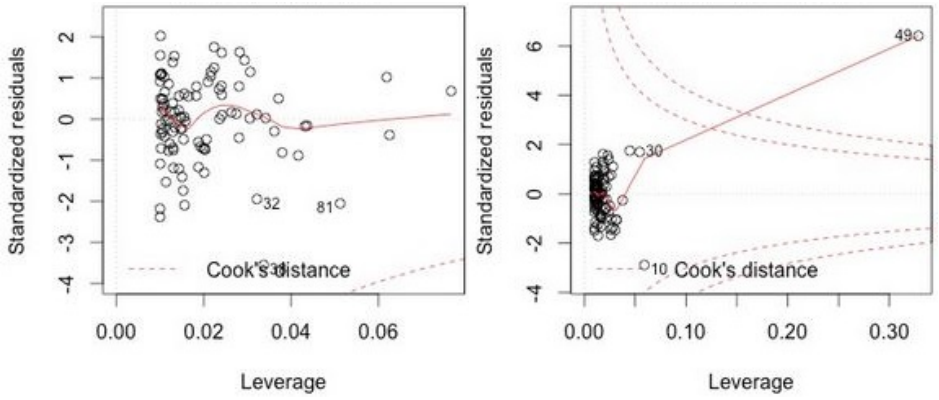


(a) Example 1 (b) Example 2

Figure 12: Scale-Location plots. Reprinted from Bommae (2015).

If there is equal variance like in Figure 12a, the smooth line is mostly horizontal. Figure 12b shows heteroscedasticity, since the variance of the residuals grows and therefore, the smooth line has a steep angle.

The Residuals vs. Leverage plot detects influential cases. Points that have a great impact on the regression line are called influential cases. Not every outlier must be an influential case, since the point can be far away from the rest but still close to the regression line. There can be points though, that change the result of the regression by a bigger margin, these can be found by a Residuals vs. Leverage plot.



(a) Example 1 (b) Example 2

Figure 13: Residuals vs. Leverage plots. Reprinted from Bommae (2015).

The plot in Figure 13b reveals one influential point in the top right (Bommae, 2015). The analysis of residuals should always be a crucial part of a regression analysis. Only with the assumptions verified, the results can be trusted.

2.4.4 Nonparametric regression

A different form of regression analysis is nonparametric regression, which makes many fewer assumptions about the data generating process. Nonparametric regression methods are local (weighted) averaging techniques and help to understand how the dependent variable Y can be explained by the independent variable X (Cameron and Trivedi, 2005),

$$Y = m(X).$$

In this case, $m(.)$ is a mathematical function and it is of interest to discover more about its specific nature. This is only a theoretical relation and might not hold for any observation, but on average it should, which is formalizd in:

$$y_i = m(x_i) + \epsilon_i,$$
$$E(Y|X = x) = m(x),$$

where $i = 1, ..., n$ and n is the number of observations. How the expectation of the variable Y conditional on X is estimated, depends on the selected method. One popular choice is kernel regression, more specifically the Nadaraya-Watson estimator:

$$\hat{m}_h(x) = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{K_h(x - X_i)}{n^{-1}\sum_{j=1}^{n} K_h(x - X_j)}\Big)Y_i = \frac{1}{n}\sum_{i=1}^{n} W_{hi}(x)Y_i.$$

K is a kernel density estimator (a generalization of the histogram estimate) and has the bandwidth h, which is responsible for the smoothness of $\hat{m}_h$ (Härdle et al., 2004). The chosen kernel has less of an impact.

Alternatively to kernel regression, the k-nearest-neighbor (k-NN) estimator is a popular choice for nonparametric regression. It simply calculates the average for the k-nearest observations. The greater k, the smoother the function.

Local polynomial regression is more flexible than standard polynomial regression. Similar to k-NN, only a part of the data is examined at one time. The width of the bandwidth, is determined by the smoothing span. Again, the greater the smoothing span, the smoother is the function.

Spline smoothing is constructed by a minimization problem

$$\hat{m}_\lambda = \arg\min_m S_\lambda(m)$$

with

$$S_\lambda(m) = \sum_{i=1}^{n}(Y_i - m(X_i))^2 + \lambda||m''||_2^2.$$

Spline smoothing includes the residual sum of squares and a punishment term that penalizes non-smoothness of $m(.)$. λ controls the weight of $||m''||_2^2$. The greater *lambda* the smoother the estimate (Härdle et al., 2004, 133).

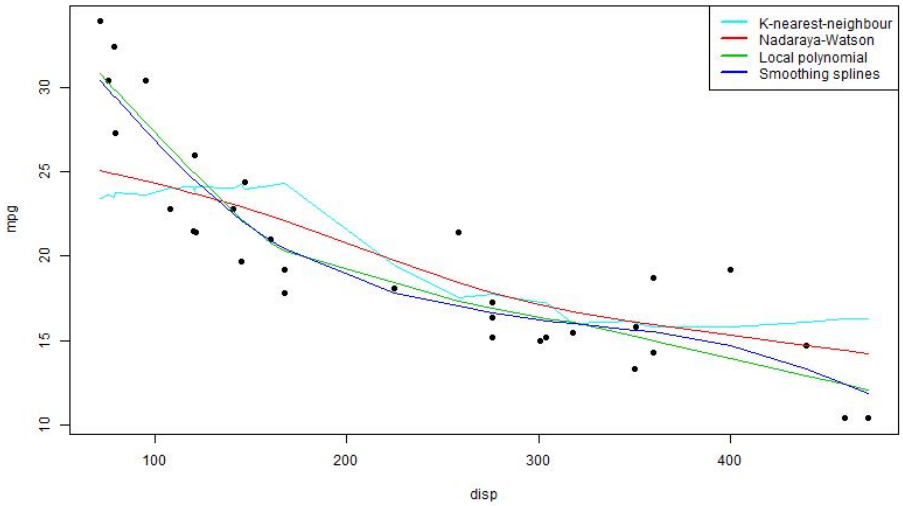


Figure 14: Nonparametric regressions of mtcars data set

Figure 20 shows an example scatter plot of two variables with four different nonparametric methods. Nonparametric models are mostly used for explorative analysis and help to identify adequate parametric models. The introduced models are flexible, but finding the optimal bandwidth or smoothing parameter is challenging.

2.4.5 Decision trees

Classification and regression trees belong to the category of decision trees. The goal is to create a model that predicts the value of a dependent variable (target), based on the information of independent variables. In classification trees, the target is a categorical variable and the tree predicts the category that the target variable is most likely to fall into. Regression trees have a continuous variable as a target and its value is predicted.

The decision tree algorithm CART (Classification And Regression Trees) was first introduced by Breiman et al. (1984). The highest node in a tree is called root node, at that point the whole data set is considered. Every split is then answering a yes-no question e.g. "Is the person *male*?" or "Is the value of *Petal.length* higher than 2.45?". The question which produces the best split, i.e. that can divide the data with the smallest error is chosen. This procedure will repeat itself, till a stopping benchmark is reached. If there is no stopping criterion in place, pruning can be done afterwards. A stopping criterion could be the amount

of observations in a terminal note, a set maximum for the depth in a tree, or a decrease in the residual of sums squared exceeding a threshold (Gareth et al., 2013). Having no such criterion could lead to overfitting. If that is the case, pruning the tree by a complexity parameter makes sense. Smaller subtrees are usually preferred. Holte (1993) even argues that *1-rule* (1R) decision trees are sufficient.

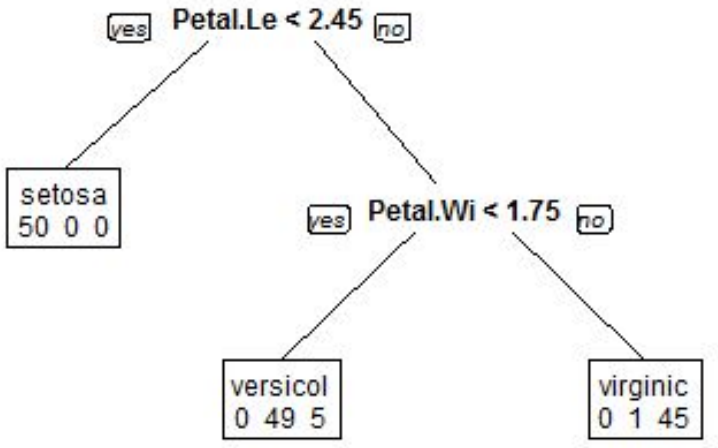


Figure 15: Classification tree of iris data set

The graph of a CART model is a binary tree, see Figure 15. The dependent variable is *Species* from the iris data set. The tree starts on top with the root node. If *Petal.Length* is lower than 2.45, all observations of the training set fall into the category *Setosa*. The second branch leads to a different node. Here *Petal.Width* determines if observations fall into the species *Versicolor* or *Virginica*. This tree provides a good set of rules on how to categorize new observations, where the species is not known. There are different ways to split nodes, for example are regression trees trying to reduce the variance. Classification trees use impurity measures like Gini index or Information gain. Results tend to be the same for the different methods (Tan et al., 2005). Advantages of classification and regression trees are the graphical display and the easy interpretation. To get better predictive accuracy, aggregating several decision trees (e.g. bagging, random forest or boosting) can be implemented (Gareth et al., 2013, 215).

3 Applications

This section will provide a short summary of statistic apps that already exist. It will then define the target group and introduce each of the newly programmed apps. The goals of each

applet are discussed and lead to a comprehensive description of its components and technical implementation. The code can be found on the attached CD or in this GitHub repository: `https://github.com/gaertnej/Shiny`. A significant part of the apps' development focused on usability and will be discussed in a summarizing chapter including all applets. The apps went through an iterative process to the final versions. Part of this process was constant qualitative feedback from master of statistics students.

3.1 Existing Apps

It is of value to look at already existing applications before focusing on the new coded **Shiny** apps. The Ladislaus von Bortkiewicz Chair of Statistics at Humboldt-Universität zu Berlin already offers a list of interactive **Shiny** statistic apps written in **R** (Härdle et al., 2015). A variety of topics include descriptive statistics, distributions, and time series analysis. This thesis provides an assortment of applets to the existing ones. It is important that newly developed apps do not already exist publicly available online. Therefore, each idea for an app needed to be researched online, so as to not duplicate any work.

Indiana State University operates the database STAT-ATTIC (STATistics Applets for Teaching Topics in Introductory Courses) with over 600 applets (DePaolo, 2015). Rossman and Chance (2005) publicize about 50 apps on their website. Often, statistics chairs like the Ladislaus von Bortkiewicz Chair offer web applications. The statistics chairs of Rice (Lane, 2008) and Berkeley University (Stark, 2013) are other examples of institutions that provide a wide collection of applets. Interactive applications can be used by students or by professors during lectures. Even though there is a large offering of programs, they might not always have the desired functionality. Customizing these apps is not trivial because code is rarely publicly available and instructors need knowledge of a variety of programming languages. Most of the presented applets are written in Java, JavaScript, HTML and CSS. The introduction of the **R** package **Shiny** in 2012, gives instructors a new possibility of creating web applications (Chang et al., 2017). The framework **Shiny** for **R** (R Core Team, 2016) only requires expertise in **R** to create interactive web applets for students. **R** is a widely used programming language for statistics instructors. Since **Shiny** is relatively new, the amount of available applets is easier to oversee. Duke University (Cetinkaya-Rundel and Cohen, 2016), Cal Poly University (Doi et al., 2016) and Albany (Dudek, 2017) offer a variety of **Shiny** apps. These are examples of available apps for lectures and students, they do not cover all available **Shiny** applets. Most of these apps concentrate on distributions and test theory. Many topics are surprisingly not

covered in-depth with **Shiny** apps yet, which is likely due to the recent release date of **Shiny**. Filling the content gap represented by the missing interactive applications will be solved over time and the following apps contribute to this process.

3.2 Target group

For the following applications, the target group are people that are learning statistical methods. Even though the apps will be publicly available, they are focusing on the usage by students from the Humboldt-Universität zu Berlin. More specifically, students who are taking introductory classes to statistics and data analysis. Students from other universities might not find specific methods in an app, if those are not covered by the Ladislaus von Bortkiewicz Chair of Statistics. The apps are supposed to be an additional learning tool for understanding. They cannot replace lectures or books about theory. The target group should understand the specific goals of the apps and use them in a compendium of learning methods.

3.3 Multivariate Graphics applet

3.3.1 Goals

The app about multivariate graphics should help students to learn different ways of visualizing multivariate data sets. Andrews curves, Parallel coordinates, Star plot, scatter plot, and Chernoff faces should be introduced and understood. As explained in Section 2.1, the order of variables has a strong impact on graphics. To understand that the order of variables changes the visualization is one goal that should be achieved by the app. In many cases the app might be the first time students see an application of Andrews curves, Parallel coordinates or Chernoff faces. Another goal is therefore the introduction to these methods and display of meaningful use. The applet should also be able to produce the problem of overplotting for Andrews curves and Parallel coordinates. Students should then be able to learn when overplotting is a problem and how to avoid it. For Andrews curves, specifically the different type of curves, provided by the package **andrews** (Myslivec, 2012) should be introduced. Further, the app has the goal of showing the influence of variable transformation for Star plot and Chernoff faces. In general, the user of the app should be stimulated to think critically about advantages and disadvantages for the different graphical solutions.

3.3.2 Components and technical implementation

The applet is constructed of a sidebar on the left, a panel for visualizations in the center, and a list panel on the right. The sidebar contains a drop-down list to select the visualization method of interest. Depending on the chosen method, the sidebar includes different input parameters. Every app also has a widget named *Data*. Clicking this widget opens a variety of options. Variables can be selected, cycled, and transformed. The data set can be changed and drawing of samples is possible.

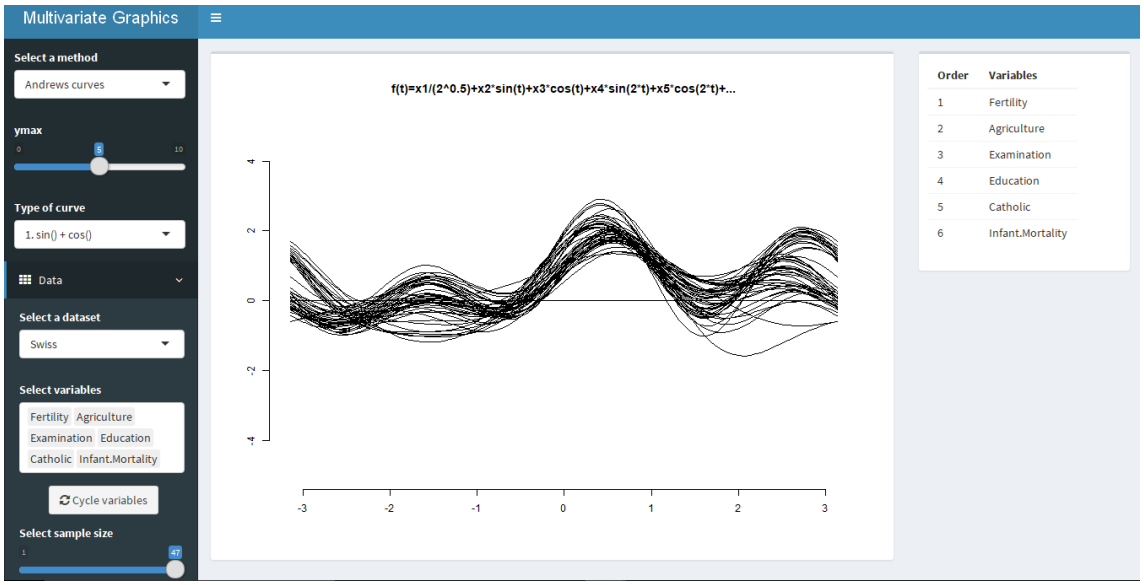


Figure 16: View of the Multivariate Graphics applet

The panel in the middle shows the selected visualization method, e.g. Andrews curves in Figure 16. The panel to the far right shows the order of the variables. The app gives the option to select out of three data sets: *USArrests*, *iris*, and *swiss*. All these data sets are part of the **datasets** package (R Core Team, 2016). The *USArrests* data contains statistics about arrests and population in all 50 U.S. states. It is chosen for its straightforward and easy interpretation for students. The *iris* flower data is often used in statistical examples and students are either familiar with it already, or can learn more about the three species. The *swiss* data set has six variables about socio-economic indicators for the French-speaking provinces in Switzerland in 1888 and provides a third example to run the methods. The selection of appropriate data sets is crucial to introduce students to meaningful applications of the discussed methods in Sections 2.1.1- 2.1.8.

Variables of the data sets are all included as a default, but can be removed. Also, the specific order of variables can be hand-selected. This and the button *Cycle variables* should

fulfill the goal to understand the influence of the order of variables in multivariate graphics.

The data set *iris* contains 150 observations and is likely to create overplotting during Andrews plots. Users of the app have the option to draw a random sample of a selected size. This should help to develop an understanding of an appropriate size for a data set for certain methods, e.g. with too many observations, Star plots and Chernoff faces might be hard to grasp.

Some methods require a certain standardization of the data, which can be learned by a transformation selection. Options are the use of *raw* data, the *z-score* and a *normalization* with range [0,1]. If the transformation of the data does not change the output, the option is not given. For a relevant use of Chernoff faces, the data must be transformed to *z-scores*, which will be the default for this method. Students can then change the data transformation and see how the plot loses its functionality. For star plots, only data with the range [0,1] creates interpretable output.

Specific parameters for the visualization methods are the following: For Andrews curves users can change *ymax* to zoom into the curves and get a closer look. Also, the four types of curves created by different fourier series can be selected. The package **stars** (R Core Team, 2016) gives the choice of drawing a radius for Star plots , which is implemented in the sidebar. The package **aplpack** (Wolf, 2014) provides a function for Chernoff faces, including different type of faces which are selectable in the app.

3.4 Categorical data applet

3.4.1 Goals

The goal of the following app is the introduction to a variety of methods to handle categorical data. The theory of these methods is presented in Section 2.1.6 and Sections 2.1.9- 2.2.2. Students should be able to understand the methods and know when and how to use them. The application will show frequency, relative frequency, and conditional frequency tables. Graphical handling of categorical data will be represented by bar charts, spine plots, mosaic plots, and parallel-sets. Users of the applet should learn the difference between stacked and grouped bar charts and the corresponding advantages and disadvantages (Section 2.1.6). For mosaic plots and parallel-sets not just the selected variables, but also their order is important. Learning this, is another objective of the app. The spine plot can only handle two categorical variables and the mosaic plot is a generalization of it. Students should be able to switch between both methods, to learn the similarities and differences. The last visual

shall introduce the Parallel coordinates for categorical data. Understanding the difference between hammock display, common-angle plots, and parallel-sets is a goal of the app. Simple frequency and relative frequency tables will be available to provide a variety of methods that give insights to categorical data. Users should also be introduced to conditional frequency tables and learn the difference between conditioning on rows and columns.

Parallel to learning various tables and graphics, a goal is the knowledge gain about association statistics. Comparing the coefficients with different data sets should develop an intuition about small and large effect sizes. Also, how association statistics are related to tables or graphics is of importance. Most important is critical thinking; students should recognize that graphics can have advantages, but sometimes a simple table is more transparent and easier to understand.

3.4.2 Components and technical implementation

Besides the sidebar on the left, the app includes one main panel which contains either a table or a graph, depending on the user input. Beneath the main panel is a panel for the association statistics, which can be hidden. The sidebar contains two radio buttons to switch between the categories of display: *graph* and *table*. Based on the selection, a drop-down menu with either the graph or table options is accessible. Selected options reveal further input parameters. A checkbox in the sidebar can hide the association statistics, but it is checked as a default. The *Data* widget unfolds variable and data selection, cycling of the variable order, and random sampling. The cycling of the variable order is integrated to understand the influence of order for mosaic plots and Parallel coordinates. The random sampling can very quickly create a variety of examples to compare association statistics. Users can decide between the *Titanic* and *HairEyeColor* data set (R Core Team, 2016). 2201 observations on four variables give information about the survival of passengers on the Titanic. Sex, hair, and eye color of 592 statistics students is provided by the *HairEyeColor* data set.

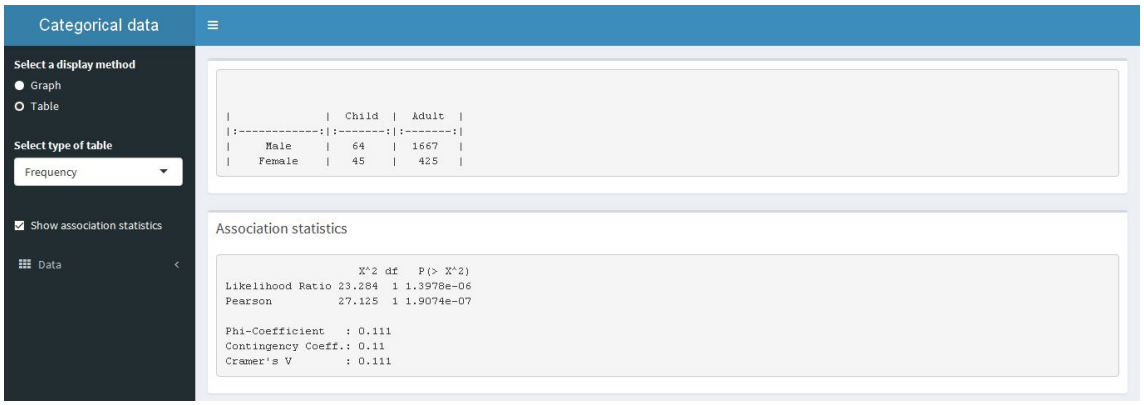


Figure 17: View of Categorical data applet

The main panel contains the selected display method, e.g. a 2x2 frequency table in Figure 17. For all tables, the package **pander** (Darczi and Tsegelskyi, 2015) was used, since it offers an appealing presentation with even four selected variables. If conditional frequency table is selected, the condition switch between rows and columns is possible through radio buttons, which should achieve a direct comparison between the two. The same principle is used if bar charts or Parallel coordinates are chosen. The different categories of plots, e.g. stacked and grouped for bar plots, or Parallel-sets, Common-angle plot, and Hammock display for Parallel coordinates, are available as radio buttons. The Parallel coordinates for categorical data are implemented with the package **ggparallel** (Hofmann and Vendettuoli, 2013).

To create a visible connection between association statistics, data tables, and visuals, the panels are available at the same time. The package **vcd** (Meyer et al., 2016) provides all statistics presented in Section 2.2. Amongst other things, students can immediately learn how the *Phi* coefficient only gives results for 2x2 tables or that *Phi* coefficient and Cramer's V have the same result in a 2x2 table. How the measures of association are related to graphics, can be best seen by the comparison between different mosaic plots and the corresponding statistics. If boxes in a mosaic plot line up mostly in a grid, only low values for contingency values are expected. The more deviation from a grid in a mosaic plot, the higher the dependence and therefore the contingency coefficient (Section 2.1.8 & Section 2.2.2).

3.5 Cluster analysis applet

3.5.1 Goals

The main goal of the app is the introduction to a variety of clustering methods. These methods consist of hierarchical, partitioning, model-based, and density-based clustering and are explained in Section 2.3. For hierarchical methods, students should learn that the result is a hierarchy of cluster solutions, which is best visualized by a dendrogram. Users of the app shall experience the effect of a variety of distance measures on the dendrogram and the cluster solution. In this context, it makes sense to show the impact of different data transformations. For agglomerative methods, also learning about the distance measures between clusters (single-linkage, average-linkage, etc.) (Section 2.3.1) must be a goal of the app. For all methods, students should not just see the result of a cluster analysis in a scatter plot, but also evaluate the cluster solution. Critical reflection on a solution will be supported by displaying silhouette plots.

One goal of the app is to clearly communicate the difference between hierarchical and partitioning methods, especially the need for prior definition of the amounts of clusters for partitioning methods as discussed in Section 2.3.2. For K-means, as the most used cluster method, different algorithms should be introduced. One important influence on partitional clustering results are the random start values, which should be highlighted by the app. Further, the app has the goal of communicating the partial membership of observations to clusters during Fuzzy clustering. For density based clustering, students should be encouraged to find the optimal ϵ and see the unique possibility to find clusters of arbitrary shape. Another goal is to showcase the effect of different values for the minimum neighbor number. In summary for this app, students should understand the workflow of each clustering method.

3.5.2 Components and technical implementation

The applet consists of a sidebar and a main panel, which holds up to three tabs. At the top, the sidebar allows for the clustering method selection in a drop-down list. Each method is listed in their corresponding sub category, e.g. hierarchical, partitioning, see Figure 18.

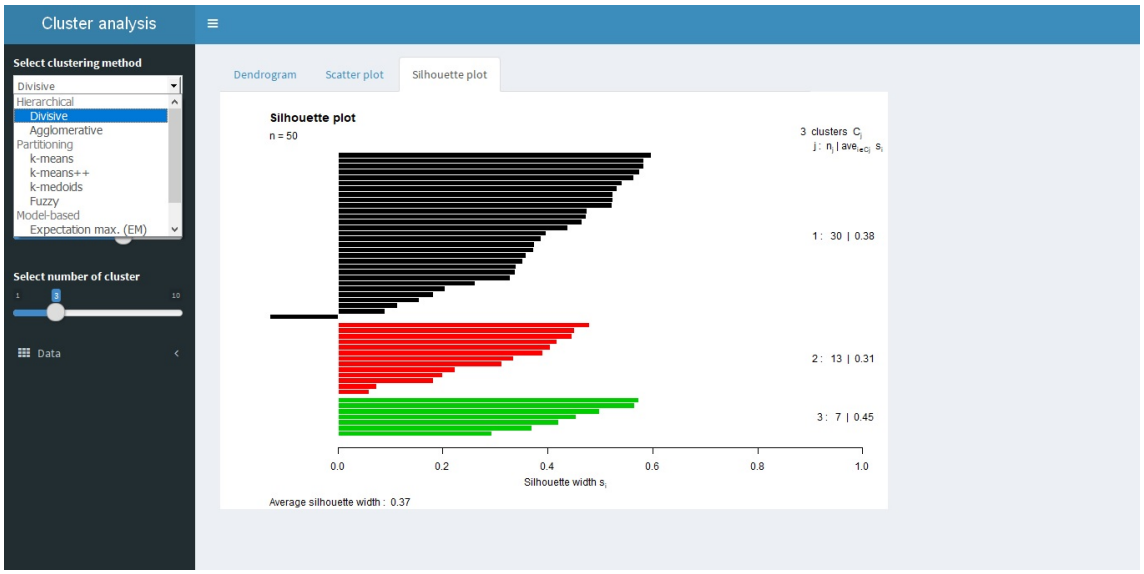


Figure 18: View of Cluster analysis applet

The chosen method reveals associated parameters in the sidebar. Besides specific parameters, for all hierarchical and partitioning methods, the number of clusters can (hierachical) or must (partitioning) be selected. The *Data* widget contains transformation options for the observations. The default is z-standardization as explained in Section 2.3. Students can experiment what happens if the data is non-standardized. The data sets inlcude *USArrrests*, *swiss* and *iris* described in Section 3.3.2. Specifically for the density-based method, the data set *multishapes* from the **factoextra** (Kassambara and Mundt, 2017) package is included. It contains multiple shapes to compare results from density-based and standard clustering methods. The *multishapes* data holds 1100 observations, which are hard for some methods to display (e.g. silhouette plot), therefore, random sampling is implemented to reduce the number of observations. All variables are selected as a default, but can be removed to analyze the influence of a single variable. The main panel consists of different tabs, which will vary through the methods. Divisive and partitional clustering hold three tabs: 1. *Dendrogram* 2. *Scatter plot* 3. *Silhouette plot*. The divisive method runs the `diana()` function from the cluster package (Maechler et al., 2016). Students can vary between euclidean and manhatten distance and observe the different looks of the dendrogram. For both hierarchical analyses the appeal of the dendrogram can be changed with the input hang, which adjusts the height of the leaves. If a number greater than 1 is selected for the amounts of clusters, the tree gets visually cut by red boxes at the corresponding height. The cluster solution then creates a scatter plot matrix in the second tab with cluster specific colored dots. How well the result performs can be evaluated in the third tab in a silhouette plot, where the bar colors corre-

spond to the colored dots. Going through the tabs simulates the process of a hierarchical cluster analysis. Starting with a hierarchy, selecting the height of the cut, analyzing the result in a scatter plot, and then evaluting it with help of a silhouette plot.

The agglomerative clustering follows the same principle, but has more options to choose from. The package **stats** (R Core Team, 2016) provides a variety of distance measures between groups, e.g. average-linkage or single-linkage. Students will observe great differences between the hierarchy of clusters in the dendrograms.

For the partitioning methods, results of the clustering solution are seen in a scatter plot matrix and the evaluation in a silhouette plots in tab number two. The `kmeans()` (R Core Team, 2016) allows the selection of the Hartigan-Wong, Lloyd, and MacQueen algorithms (Section 2.3.2). As for all other partitioning methods, the number of clusters K has to be chosen before a result is calculated. The default for K is 1 (which will not give a solution), to stimulate a conscious decision process. A button for recalculating ensures the understanding of different start values. For the K-medoid method the start values are picked by the `pam()` function from the **cluster** package (R Core Team, 2016) in the so called build phase. They are therefore always the same and a recalculation button was not implemented. For Fuzzy clustering a third tab was added, to provide a table for the degrees of membership. The function `cmeans()` from the **e1071** package (Meyer et al., 2017) lets the user change the degree of fuzzification, which is also implemented as a parameter in the app.

The model-based clustering was integrated with the `Mclust()` function from the correspondent package (Fraley et al., 2017). The optimal cluster solution is picked by the function, but students can follow the decision process on the first tab, which displays the BIC for multiple K. For the density-based method of clustering, the `dbscan()` function from the **dbscan** package (Hahsler and Piekenbrock, 2017) is used. The function is all about selecting an optimal ϵ. Students are assisted in finding a fitting ϵ on the first tab. The result can then be evaluated in the scatter plot and the silhouette plot. The app guides the student through a variety of cluster methods and showcases different ways of finding an appropriate cluster solution.

3.6 Linear regression applet

3.6.1 Goals

The applet about linear regression is supposed to introduce different models to students, ranging from binary regression and polynomial regression to multiple regression with inter-

action. How the model forms a regression line, should be displayed in a scatter plot. One of the app goals is that users recognize the difference in flexibility for different models. Students should learn the advantages of two- and three-dimensional scatter plots for models with two independent variables. Besides the regression line or plane, students should also be able to learn more about the model itself. The app needs to provide the parameter estimates and corresponding t-tests. Also, a comparison between R^2 and adjusted R^2 needs to be possible. Another goal is to develop familiarity for students with the output provided by **R**. For all models, users should be able to analyze the residuals, check if the assumptions (Section 2.4.3) are violated and think about following consequences.

3.6.2 Components and technical implementation

A main panel with three tabs and a sidebar construct the applet. In a drop-down menu, the regression model can be chosen. The options are the simple regression model with and without constant, models of second and third polynomial order, and multiple regression models containing two variables with and without interaction. Since the variables selected are an essential part of the regression model, the variables are not hidden in the *Data* widget, but beneath the model selection. The *Data* widget contains just the variable transformation for simple regression without a constant because only then can data transformation from raw data to *z-score* have an impact. The available data sets are *mtcars*, *iris*, and *swiss*. There are eleven variables about automobile design and performance in 32 observations in the *mtcars* data. To vary the observations, there is again an implementation for random sampling.

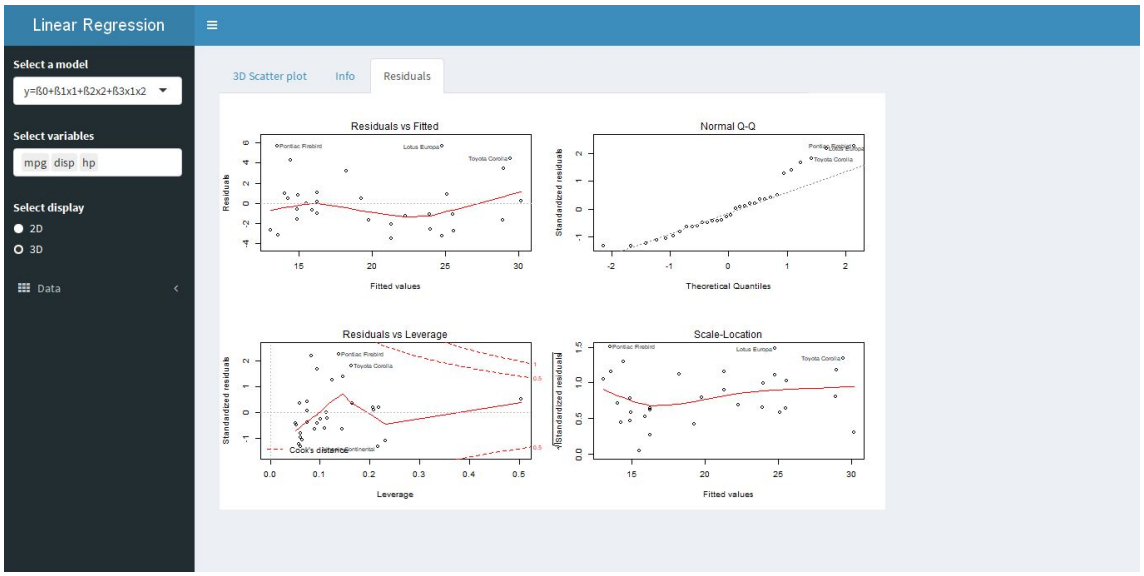


Figure 19: View of Linear regression applet

The main panel consist of three tabs: 1. *Scatter plot* 2. *Info* 3. *Residuals*, see Figure 19. The scatter plot tab lets students see the observations with the regression line. If it is a three-dimensional scatter plot, created by the rendering package **rgl** (Adler and Murdoch, 2017), the user can rotate the plot. This will allow for a view of the regression plane from all angles. The info tab shows the classical regression summary output provided by **R**. It displays the regression estimates, the corresponding t-tests, R^2 and adjusted R^2, plus additional information about the model. Students are therefore able to learn, to grasp the most important information from a **R** regression output. The last tab shows four residual plots explained in Section 2.4.3 at a time. These plots are available with the `plot()` function of a regression model. Users of the app can quickly compare different regression models and evaluate the quality of the model. Further, they become familiar with the **R** output of regression models and residual plots.

3.7 Nonparametric regression applet

3.7.1 Goals

The main goal of this app is to show the influence of parameters on the different nonparametric regression methods. Besides learning the dynamic of the methods themselves, a comparison of the methods should also be possible. The applet will introduce k-nearest-neighbor, Nadaraya-Watson, local polynomial, and smoothing splines regression. Each method, as introduced in Section 2.4.4, has at least one parameter that changes the smoothness of the regression line. Students should know after using the app how the regression line changes depending on the alteration of the parameter. This should lead to an intuitive understanding of nonparametric regression.

3.7.2 Components and technical implementation

The app consists of a sidebar on the left and a main panel in the center. The sidebar holds the four nonparametric regression methods, k-nearest-neighbor, Nadaraya-Watson, local polynomial, and smoothing splines, which are selectable by checkboxes.

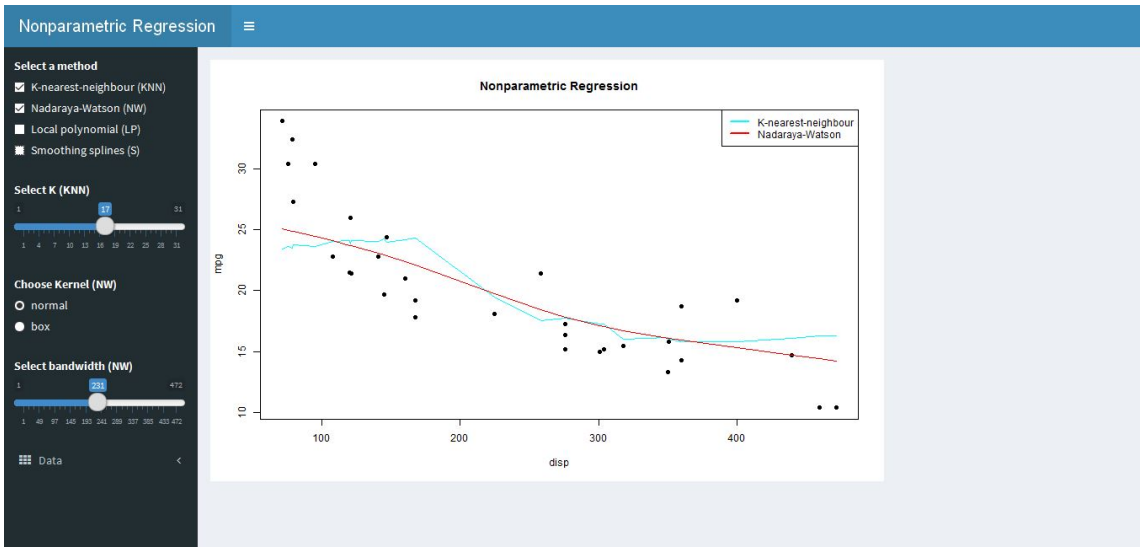


Figure 20: View of Nonparametric regression applet

If a method is selected, the corresponding input parameter appears in the sidebar, e.g. if Nadaraya-Watson is selected, the kernel and the bandwidth can be changed (Figure 2.4.4). Students can choose between the *cars* and the *swiss* data set in the *Data* widget. Also, for a variety of examples, the dependent and independent variables can be changed. To reduce the number of observations, random sampling is implemented. The main panel holds only a scatter plot. It can show all regression lines at the same time and differentiates them by color. By changing the parameters in the sidebar, students immediately see the reshaping of the regression line. It is then obvious how increasing k smooths the function of k-nearest-neighbor or how the kind of kernel has less influence than the bandwith in Nadaraya-Watson regression. The **stats** package (R Core Team, 2016) holds the function `lowess()` for local polynomial regression, `ksmooth()` for Nadaraya-Watson, and `smooth.spline()` for smoothing splines. `Lowess()` allows the change in the smoother span, `smooth.spline()` includes a smoother parameter which calculates λ, and `ksmooth()` lets the user change the kernel and the bandwith. For k-nearest-neighbor, the function `knn.reg()` from the **FNN** package (Beygelzimer et al., 2013) allows for the variation of k. All these options are integrated as parameters in the app. A variety of parameters can be tried out by students to observe the effect of nonparametric regression.

3.8 Decision trees applet

3.8.1 Goals

The primary objective of this app is to introduce classification and regression trees. Students should be able to understand the trees and the corresponding **R** output about the solution. Also, different cut offs should be available, like the maximal depth of the tree or a complexity parameter (Section 2.4.5). To change the result of a tree, a parameter for the minimal number of observations in a node should be available as well. The user of the app should learn that pruning can create the same result as defining the maximal depth of a tree, the only difference being the approach to get there. Using pruning should also make the user think about the advantages of smaller trees. Besides the trees, the app also aims to build familiarity with the output of the `summary()` function of a decision tree. Which also gives the option to learn more about the difference between Gini index and Information gain for classification trees. The app should be a tool to explore the functionality of decision trees.

3.8.2 Components and technical implementation

The app about decision trees has a sidebar on the left and a main panel containing two tabs. Students can choose between regression and classification trees at the top of the sidebar. The decision will affect the available data sets. To answer different questions of interest, the target variable is changeable in a drop-down list. To user can then decide if pruning by a complexity parameter should be done or if the size of the tree should be determined by the maximal depth. If the checkbox for pruning is not selected, a parameter for maximal depth is on display with a default setting of 1, which is equivalent to the 1R approach (Section 2.4.5). If pruning is selected, the complexity parameter appears and has a default setting of 0, which causes the construction of the full tree. By then increasing the complexity parameter, students see how the tree shrinks to a reasonable size. The size of the tree can also be changed by varying the minimum amount of observations in a node. For classification trees, two radio buttons are available to change between the criterion for classification. The *Data* widget in this app only holds different data sets and the option for random sampling. The data sets for classification trees are *Titanic* and *iris*. The *iris* data set only contains one categorical variable, which is therefore, automatically the target variable. For regression trees the data sets *mtcars*, *swiss* and *CPS85* are selectable. *CPS85* is from the package **mosaicData** (Pruim et al., 2016), contains information about the Current Population Survey from 1985 and is reduced to the variables *wage*, *education*, and *experience* for clarity.

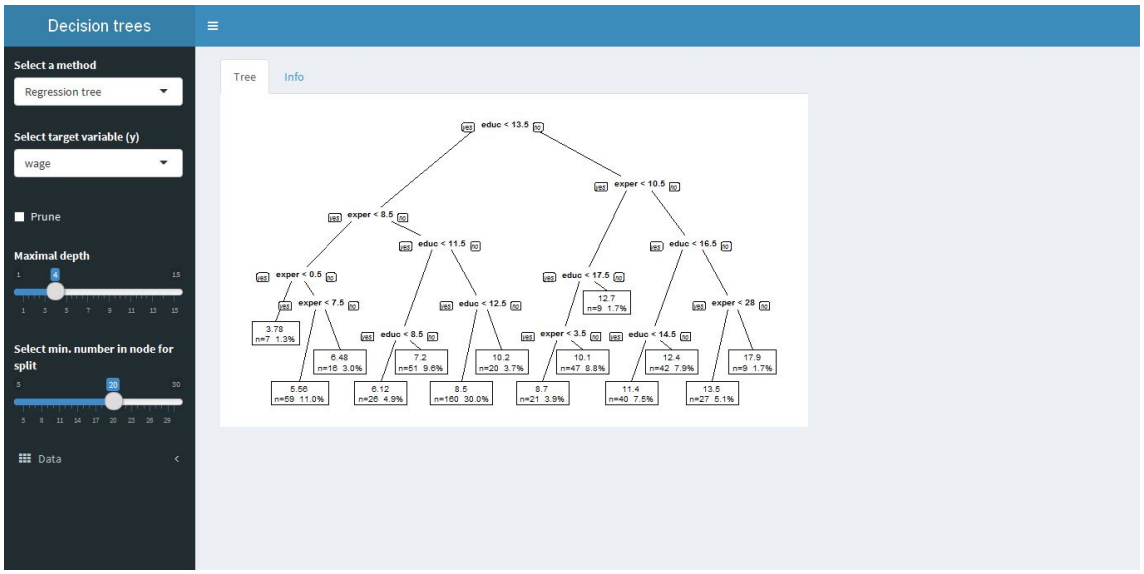


Figure 21: View of Decision trees applet

The main panel shows the tree in the first tab and more information about the model in tab two. The tree model is built with the `rpart()` function from the correspondent package (Therneau et al., 2015). The authors created the function based on the ideas of Breiman, Friedman, Olshen, and Stone (1984) (Section 2.4.5). There are several packages on plotting decision trees and **rpart.plot** (Milborrow, 2017) - arguably the most structured and readable package - was implemented, see Figure 2.4.5. The info tab gives student the chance to access a more in-depth analysis of the trees. They can see at which complexity parameter (CP) value the tree gets pruned. Also, the difference between the information criterions becomes apparent. The app with all its options is a good starting point for students to experiment with practical applications of decision trees.

4 Usability

After establishing the goals and desired functionality, the design of the app must be built. The development of an app with a logical flow and an attractive interface is an iterative process. More about the process and usability tests will be presented in the next chapter. The importance of usability in a web application cannot be overstated. Programming is only successful if an average person can figure out how to use the app (Krug, 2014). Even if all functions are integrated, the application is unsuccessful if students do not understand how to use it. Users of the app are most likely going to be students at Humboldt-Universität zu Berlin, who are taking introductory classes to statistics or data analysis. The apps are

not trying to substitute lectures and therefore, lengthy explanations about the methods are missing. Users will need to have a basic understanding of the corresponding theory to make sense of the display and available parameters. Since users do not read, but rather scan websites (Krug, 2014, 30–35), any sort of explanations are reduced to a minimum. For an effortless handling of a web application, it is suggested by (Tidwell, 2011) to use modern website conventions. Even if there are solutions that are just as good, users will be more comfortable with a design they are familiar with. Following these conventions, the title of the app is placed in the top left and the parameters are located in the sidebar on the left. The center of the app holds the content that should receive the most attention from the user. In the newly introduced applets, this is either a graph, a table, or **R** text output. Output that belongs together is organized in tabs in the center of the screen. Tabs should be used when contents are related to each other, but do not have to be seen at the same time. They are straightforward in their use and cause no confusion (Tidwell, 2011, 157-158). Usability tests of the apps showed that tabs were less often overseen, in comparison to parameters in the sidebar. They also reduce the risk of an overloaded sidebar, which contains too many options. The apps about categorical data and multivariate graphics contain no tabs, but have more than one panel. In both cases, the second panel is not just related to the main panel, but aids comprehension the output, e.g. the order of variables is displayed next to the Andrews curves in Figure 16.

The reasoning behind the selection of parameters is explained in Sections 3.3- 3.8.2. For each app, a method can be picked by the user. In most cases, a drop-down list is used as an input parameter. It uses relatively little space, it can include headers, and shows all methods if clicked on. A downside is that not all options can be seen at the same time, but this is outweighed by the benefits. If several methods can be selected at the same time, checkboxes are used (e.g. Nonparametric regression applet, Figure 20). They do require more space, but display all options at the same time and imply multiple selections (Tidwell, 2011, 345-346). Students often have the chance to change numerical values (number of clusters, sample size, bandwidth etc.). A slider proved to be the best solution to select numerical values, since a minimum and maximum value is fixed, and a default value can be programmed. The tick values are presented by **Shiny** in a very crowded way. Ticks are therefore hidden by recommendation of students in usability tests. Users also complained about the fact that the numerical value cannot be clicked on in the bar. The **Shiny** slider only allows for dragging, to select the appropriate number. It could not be manipulated at the time, but might be

implemented by the **Shiny** team in the near future. If space is not a problem in the sidebar, parameter options are selectable with radio buttons. For variable selection, more than one variable is often selectable. **Shiny** provides the function `selectInput()` to select multiple items at a time, but hides all options if the parameter is not clicked on. It does save space, but caused confusion during usability tests. Several times, it was unclear to students how to delete already selected variables. A tooltip proved to be sufficient to explain the correct use of the input selection. Every app contains a *Data* widget, which reveals a series of options to change the data. It was implemented to save space and hide parameters that are not directly associated with the selected method. Since the data parameters have an impact on the output, it was still important to include them. Usability tests showed that all parameters were available to students and the concept was self-explanatory. Each app also contains buttons, either for sampling or to re-run a method. Refresh icons are put on the buttons, to facilitate the function.

To create user-friendly apps, tips and help statements must be implemented. If input parameters cause error messages in **R**, programmers might be able to understand what causes the problem, but the applets should not expect that depth of knowledge from students. Therefore, each possible error message is replaced by a help statement, which gives exact instructions on how to fix the error e.g. "Select at least two variables". Sometimes plots or parameters need a little explanation. These tips are implemented with the `bsTooltip()` function from the **shinyBS** package (Bailey, 2015). They only appear while hovering over the corresponding component of the app. This avoids lengthy explanations and guarantees a clean look. How and which components benefit from tool tips was established through usability testing.

The apps are built to generate a logical flow. Parameters on the top correspond to decisions that must be made before parameters further down. At the beginning of every process is the decision about which method to choose. The selected method is responsible for which parameters show up. These are ordered in a linear flow for a more natural use (Reiss, 2012). To hide parameters the function `conditionalPanel()` is carried out. The general theme for designing the apps is "don't make people think", preached by Krug (2014) regarding websites and applications. Usability testing contributed greatly to the development of self-explanatory, user-friendly apps.

4.1 Usability tests

Collecting empirical data during the observation of representative users using a product is referred to as usability testing. There are different approaches to usability testing. One approach is a more formal test, trying to confirm or reject a hypothesis. The other method is less formal, but still very detailed and tries to form a product through an iterative process, to improve weaknesses by testing multiple times (Rubin and Dana, 2008). The second approach was used to shape the presented apps from Section 3. Since usability is difficult to measure, only qualitative data was collected. Testing was done multiple times, starting around two months into the project with five master of statistics students. One-on-one sessions with students lasted about one hour each. After the introduction of the apps, testing times decreased significantly in later sessions. Users were asked to discover the functionality of the apps and think aloud. Thinking aloud is one of the most popular methods for usability testing. It is a cheap and reliable way to gain insights about the cognitive processes of human behavior (Nielsen, 1993). The situation can feel unnatural for the user, but the method proved to be invaluable. Usability testing revealed a couple of simple errors, which had been previously overlooked. This could be misspelling or a method not working appropriately. The greatest effect of usability testing was the addition of tooltips and error messages including recommmendations. Often users got stuck in the app, but understood the logic after a brief explanation. For example, it was unclear how to delete variables in a parameter that allowed the selection of several variables. The tooltip "Select and press delete to remove variable" proved to be sufficient for understanding the use of the parameter. Also, error messages by **R** did not give any advice on how to fix the corresponding problem, which is why they had to be replaced by alternative ones. Working on these changes improved the understanding of the apps tremendously. Testing not only lead to adding explanations, but also triggered the removal of parameters. The paring down of options constituted a balancing act between maintaining all relevant parameters and not overwhelming users with too many options. A parameter to change the maximum iterations for different clustering methods was removed, since it caused more confusion and had no significant impact on the result. The default value is now used for each method and the app is less crowded. Even an entire app was deleted after usability testing. An app about neural networks revealed itself as not efficient enough in imparting knowledge. It was therefore removed and it exposed how important a fitting topic and well-defined goals are for app development. Having users test the app also showed which parts of the app should stay the same. If the logical flow was well understood and no

explanations were necessary, there was no reason for alteration. Getting feedback from the students after the first testing sessions uncovered the interest in more theoretical background on the methods. None of the test subjects were concurrently enrolled in classes covering any of the methods, but all had learned them previously. Suggestions for theoretical background knowledge provided by the app were noticed but not followed up because of the set goals for the apps. Future improvements could include links or short explanations about the theory in a separated tab. The goal for all apps was to be a complementary tool for lectures. To make them all-encompassing, including theory, practice, and possibly even test questions could be accomplished in later versions. Real use demonstrations and evaluation were invaluable to the look and functionality of the apps. Tester feedback improved the logic and overall quality of the **Shiny** applications.

5 Conclusions

Teaching statistics is a complex task and should be approached from different angles. The applications introduced here give students the opportunity to explore a wide variety of statistical methods. Equivalent apps could not be found online, which makes these applets unique in the way they present a specific topic. The applets are an addition to the collection from the Ladislaus von Bortkiewicz Chair of Statistics. They are therefore tailored for students taking classes of the Humboldt-Universität zu Berlin Statistics Chair. Lectures are densely packed with theoretical knowledge and do not guarantee that students understand the transferred input immediately. These Shiny applets are another tool available for students to become more proficient with the curriculum content and a complement to lectures and slide review. The web applications allow students to test different specifications or models and compare the corresponding results. This can lead to an intuitive understanding of the dynamics and influence of parameters. Since the apps are even available on mobile devices and no programming is necessary, it is expected that students will engage with the offer. Students' continued use of the apps should be evaluated by the statistics chair in the future.

To assure the understanding of the apps, feedback from fellow master of statistics students was crucial. It showed that the newly developed apps were experienced as more complex than the already existing ones, which is also reflected in the greater number of input parameters. The evaluation of the applets led to an increase in helpful messaging, tooltips, and deletion of parameters. Usability testing repeatedly revealed that students wanted theoretical input in the app itself, even if it were "just a couple of sentences". This improvement could be made in future versions. After the release of the apps, current students for whom the applets are designed, should be evaluating them. Those usability tests or feedback rounds will lead to even better insights than the previous evaluation.

In a next step, it has to be assessed how the web applications can be further utilized. Again, the GAISE report from 2016 can be referenced as a guideline, created by experts about statistics education. The apps mostly target the recommendation to use technology for exploring concepts and analyzing data, but they also have the potential to fulfill even more recommendations, if correctly designed and used. One relatively easy improvement of all apps would be the option to upload customized data. To "integrate real data with a context and a purpose" is a recommendation that can lead to more engaged students and appreciation of the course material (Neumann et al., 2013). The data sets used so far are standard examples in the statistics community and could be perceived as irrelevant to students. It is even suggested

to make students collect their own data, if time is available. Simple data sets could be easily uploaded and explored, e.g. in the applet about categorical data.

To accomplish even better results from the apps, questions for students can be integrated. Those questions can be generic and challenge students' conceptual knowledge, or very specific and test the students' ability to use the correct method in a way to find the answer. The corresponding feedback by the applet is beneficial for students and professors. Misconceptions are revealed and give professors the opportunity to solve them. Also, students get the chance to test their understanding of the methodology. To improve students learning, providing such feedback is recommended by GAISE (2016). Questions can also tackle the recommendations to: 1. "Teach statistical thinking" and 2. "Focus on conceptional understanding". By asking about interpretations and implications of results, students can develop a deep understanding of concepts. Also, critical thinking can be motivated with questions, for example about advantages and disadvantages of certain statistical methods. Questions can be asked in an open format, to trigger thought processes or in a multiple-choice format, to provide immediate feedback for students and the chance to reveal misconceptions.

With the introduction of **Shiny**, the question about the feasibility of statistics applets (Härdle et al., 2007) is solved. Only programming knowledge in **R** is necessary, which most statistics professors are already familiar with. Every script and method can easily be transformed into a web application. It is to be expected that most relevant statistical methods will soon have a corresponding **Shiny** application. The value of usability tests and students' feedback cannot be underestimated. How much students are involved in the development of those learning tools will determine their success.

When students become *statistical thinkers*, statistics education was successful and statistic applets can help to reach that goal.

A Appendix

List of files on CD

Applet_Categorical_data.r

Applet_Cluster_analysis.r

Applet_Decision_trees.r

Applet_Linear_regression.r

Applet_Multivariate_graphics.r

Applet_Nonparametric_regression.r

Gärtner_thesis.pdf

References

Adler, D. and D. Murdoch (2017). *rgl: 3D Visualization Using OpenGL*. R package version 0.98.1 URL `https://CRAN.R-project.org/package=rgl`.

Andrews, D. (1972). Plots of high-dimensional data. *Biometrics*, 28: 125-136.

Arthur, D. and S. Vassilvitskii (2007), *K-means++: The Advantages of Careful Seeding*, Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, 1027–1035.

Bailey, E. (2015). *shinyBS: Twitter Bootstrap Components for Shiny*. R package version 0.61 URL `https://CRAN.R-project.org/package=shinyBS`.

Beygelzimer, A., S. Kakadet, J. Langford, S. Arya, D. Mount, and S. Li (2013). *FNN: Fast Nearest Neighbor Search Algorithms and Applications*. R package version 1.1 URL `https://CRAN.R-project.org/package=FNN`.

Breiman, L., J.H. Friedman, R.A. Ohlsen, and C.J. Stone (1984). *Classification and Regression Trees*. Wadsworth, Inc, Monterey, Calif.

Bommae, K. (2015). Understanding Diagnostic Plots for Linear Regression Analysis. University of Virgina Library. URL `http://data.library.virginia.edu/diagnostic-plots/`. Last accessed on Apr 20, 2017.

Bortz, J. and C. Schuster (2010). *Statitik für Human- und Sozialwissenschaftler*. Springer-Verlag, Berlin Heidelberg, 7th edition.

Cameron, A.C. and P.K. Trivedi (2005). *Microeconometrics. Methods and Applications*. Cambridge University Press, Cambridge.

Cetinkaya-Rundel, M. and B. Cohen (2016). ShinyEd. URL `http://www2.stat.duke.edu/~mc301/shinyed/`. Last accessed on Mar 05, 2017.

Chambers, J.M., W.S. Cleveland, B. Kleiner, and P.A. Tukey (1983). *Graphical methods for data analysis*. Wadsworth International Group, Belfort, Calif.

Chance, B., D. Ben-Zvi, J. Garfield, and M. Elsa (2007). The Role of Technology in Improving Student Learning of Statistics. *Technology Innovations in Statistics Education*, 1(1).

Chang, W., J. Cheng, J.J. Allaire, Y. Xie, and J. McPherson (2017). *shiny: Web Application Framework for R*. R package version 1.0.0 URL `https://CRAN.R-project.org/package=shiny`.

Chen, C., W. Härdle, and A. Urwin (2008). *Handbook of Data Visualization*. Springer-Verlag, Berlin Heidelberg.

Chernoff, H. (1973). The Use of Faces to Represent Points in K-Dimensional Space Graphically. *Journal of the American Statisical Association*, 68: 361-368.

Cochran, W.G. (1952). The χ^2 test of fit. *Annals of Mathematical Statistics*, 25: 315-345.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum, Hillsdale, NJ, 2nd edition.

Darczi, G. and R. Tsegelskyi (2015). *pander: An R Pandoc Writer*. R package version 0.6.0 URL `https://CRAN.R-project.org/package=pander`.

DePaolo, C. (2015). STAT-ATTIC, STATistics Applets for Teaching Topics in Introductory Courses. Indiana State University. URL http://sapphire.indstate.edu/~stat-attic/index.php. Last accessed on Mar 05, 2017.

Doi, J., G. Potter, J. Wong, I. Alcaraz, and P. Chi (2016). Web Application Teaching Tools for Statistics Using R and Shiny. Cal Poly. *Technology Innovations in Statistics Education*, 9(1). URL https://statistics.calpoly.edu/shiny.

Dudek, B. (2017). R/Shiny Applications for Learning Statistical Concepts. University at Albany Psychology Department. URL http://www.albany.edu/psychology/statistics/shinypsych.htm. Last accessed on Mar 05, 2017.

Embrechts, P. and A.M. Herzberg (1991). Variations of Andrews Plots. *International Statistical Review*, 59(2): 175–194.

Ester, M., H. Kriegel, J. Sander, and X. Xu (1996), *A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. AAAI Press, 226-231.

Everitt, B.S., S. Landau, M. Leese, and D. Stahl (2011). *Cluster analysis*. John Wiley & Sons, Inc., London, 5th edition.

Fraley, C., A.E. Raftery, T.B. Murphy, and L. Scrucca (2012). mclust Version 4 for R: Normal Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. https://www.stat.washington.edu/research/reports/2012/tr597.pdf. Last accessed on May 10, 2017.

Fraley, C., A.E. Raftery, T.B. Murphy, L. Scrucca, and M. Fop (2017). *mclust: Gaussian Mixture Modelling for Model-Based Clustering, Classification, and Density Estimation*. R package version 5.3 URL https://CRAN.R-project.org/package=mclust.

Friendly, M. (1991). Statistical Graphics for Multivariate Data. Paper presented at the SAS SUGI 16 Conference, URL http://uregina.ca/~gingrich/text.htm. Last accessed on Apr 06, 2017.

Friendly, M. (1994). Mosaic displays for multi-way contingency tables. *Journal of the American Statistical Association*, 89: 190–200.

Friendly, M. (2006), *A brief history of data visualization*, Handbook of Data Visualization. Springer-Verlag, 15–57.

Garca-Osorio, C. and C. Fyfe (2005). Visualization of High-Dimensional Data via Orthogonal Curves. *Journal of Universal Computer Science*, 11: 1806–1819.

Gareth, J., D. Witten, T. Hastie, and R. Tibshirani (2013). *An Introduction to Statistical Learning*. Springer-Verlag, New York.

GAISE (2005). Guidelines for Assessment and Instruction in Statistics Education - College Report. *Amercian Statistical Association*. URL http://www.amstat.org/asa/files/pdfs/GAISE/2005GaiseCollege_Full.pdf. Last accessed on May 16, 2017.

GAISE (2016). Guidelines for Assessment and Instruction in Statistics Education (GAISE) - College Report 2016. *Amercian Statistical Association*. URL http://www.amstat.org/asa/files/pdfs/GAISE/GaiseCollege_Full.pdf. Last accessed on Jun 16, 2017.

Gentle, J.E. (2002). *Computational statistics*. Springer-Verlag, New York.

Gingrich, P. (2004). Introductory Statistics for Social Sciences. URL `http://uregina.ca/~gingrich/text.htm`. Last accessed on May 15, 2017.

Hahsler, M. and M. Piekenbrock (2017). *dbscan: Density Based Clustering of Applications with Noise (DBSCAN) and Related Algorithms*. R package version 1.0-0 URL `https://CRAN.R-project.org/package=dbscan`.

Härdle, W., M. Müller, S. Sperlich, and A. Werwatz (2004). *Nonparametric and Semiparametric Models*. Springer-Verlag, Berlin Heidelberg.

Härdle, W., S. Klinke, and W. Ziegehagen (2007). On the Utility of E-Learning in Statistics. *International Statistical Review*, 75(3): 355-364.

Härdle, W., S. Klinke, and B. Rönz (2015). *Introduction to Statistics: Using Interactive MM*Stat Elements*. Springer International Publishing, Berlin Heidelberg. URL `http://mars.wiwi.hu-berlin.de/mediawiki/mmstat3/index.php/Liste_interaktiver_Beispiele`.

Hinum, K.A. (2006). Gravi++ -An Interactive Iinformation Visualization for High Dimensional, Temporal Data. PhD thesis, Vienna University of Technology, Faculty of Informatics.

Hofmann, H. and M. Vendettuoli (2013). Common angle plots as perception-true visualizations of categorical associations. *IEEE Transactions on Visualization and Computer Graphics*, 19(12): 2297–2305.

Holte, R.C. (1993). Very Simple Classification Rules Perform Well on Most Commonly Used Datasets. *Machine Learning*, 11: 63–91.

Howell, D.C. (2011). Chi-square test: analysis of contigency tables. *International Encyclopedia of Statistical Science*: 250–252.

Inselberg, A. (1997). Multidimensional detecitve. *Information Visulaization, 1997. Proceedings., IEEE Symposium on*: 100–107.

Inselberg, A. (2009). *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. Spiegel-Verlag, Secaucus, NJ.

Kassambara, A. and F. Mundt (2017). *facoextra: Extract and Visualize the Results of Multivariate Data Analyses*. R package version 1.0.4 URL `https://CRAN.R-project.org/package=factoextra`.

Kaufman, L. and P.J. Rousseeuw (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Inc., Hoboken, NJ.

Kosara, R., F. Bendix, and H. Hauser (2006). Parallel Sets: interactive exploration and visual analysis of categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(4): 558-568.

Krug, S. (2014). *Don't Make Me Think, Revisited, A Common Sense Approach to Web Usability*. New Riders, United States of Amerika, 3rd edition.

Lane, D.M. and Z. Tang (2000). Effectiveness of simulation training on transfer of statistical concepts. *Journal of Educational Computing Research*, 22(4): 383-396.

Lane, D.M. (2008). Rice Virtual Lab in Statistics. URL `http://onlinestatbook.com/rvls/index.html`. Last accessed on Mar 02, 2017.

Lanzenberger, M. (2003). The Interactive Stardinates - An Information Visualization Technique Applied in a Multiple View System. PhD thesis, Vienna University of Technology, Faculty of Science and Informatics.

Liebetrau, A.M. (1983). *Measures of Association*. SAGE Publications, New York.

Lloyd, S.P. (1982). Least squares quantization in PCM. Speacial issue on quantiuation, 28: 129-137.

Maechler, M., P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik (2016). *cluster: Cluster Analysis Basics and Extensions*. R package version 2.0.5 URL `https://CRAN.R-project.org/package=cluster`.

Manning, C.D., R. Prabhakar, and H. Schütze (2008). *Introduction to Information Retrieval*. Cambridge University Press, New York.

McDaniel, S. and L. Green (2012). Using Applets and Video Instruction to Foster Students‘ Understanding of Sampling Variability. *Technolofy Innovations in Statistics Education*, 6(1).

Meyer, D., A. Zeileis, and K. Hornik (2003). Visualizing Independence Using Extended Association Plots. *Proceedings3rd International Workshop on Distributed Statistical Computing*: 1–8.

Meyer, D., A. Zeileis, and K. Hornik (2016). *vcd: Visualizing Categorical Data*. R package version 1.4-3 URL `https://CRAN.R-project.org/package=vcd`.

Meyer, D., E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch (2017). *e1071: Misc Functions of the Department of Statistics, Probability*. R package version 1.6-8 URL `https://CRAN.R-project.org/package=e1071`.

Milborrow, S. (2017). *rpart.plot: Plot rpart Models: An Enhanced Versino of plot.rpart*. R package version 2.1.1. URL `https://CRAN.R-project.org/package=rpart.plot`.

Mohamad, I.B. and D. Usman (2013). Standardization and Its Effects on K-Means Clustering Algorithm. *Research Journal of Applied Sciences, Engineering and Technology*, 6(17): 3299-3303.

Morissette, L. and S. Chartier (2013). The k-means clustering technique: General considerations and implementation in Mathematica. *Tutorials in Quantitative Methods for Psychology*, 9(1): 15-24.

Myslivec, J. (2012). *andrews: Andrews curves*. R package version 1.0. URL `https://CRAN.R-project.org/package=andrews`.

Neumann, D., M. Hood, and M. Neumann (2013). Using Real-Life Data when Teaching Statistics: Student Perceptions of this Strategy in an Introductory Statistics Course. *Statistics Educatio Research Journal*, 12: 59–70.

Nielsen, J (1993). *Usability Engineering*. Morgan Kaufmann, San Francisco.

Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, 50(302): 157-175.

Pruim, R., D. Kaplan, and N. Horton (2016). *mosaicData: Project MOSAIC Data Sets*. R package version 0.14.0 URL `https://CRAN.R-project.org/package=mosaicData`.

Rawlings, J.O., S.G. Pantula, and D.A. Dickey (1998). *Applied Regression Analysis: A Research Tool*. Springer-Verlag, New York, 2nd edition.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Reiss, E. (2012). *Usable Usability, Simple Steps for Making Stuff Better*. John Wiley & Sons, Inc., Indianapolis.

Rossman, A.J. and B. Chance (2005). Rossman/Chance Applet Collection. URL `http://www.rossmanchance.com/applets/`. Last accessed on Mar 02, 2017.

Rousseeuw, P.J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20: 53-65.

Rubin, J. (1967). Optimal classification into groups: an approach for solving the taxonomy problem. *Journal of Theoretical Biology*, 15: 103-144.

Rubin, J. and C. Dana (2008). *Handbook of Usability Testing*. Wiley Publishing, Inc., Indianapolis.

Schlittgen, R. (2004). *Das Statistiklabor: Einfhrung und Benutzerhandbuch*. Springer-Verlag, Berlin Heidelberg.

Schlittgen, R. (2009). *Multivariate Statistik*. Oldenbourg Wissenschaftsverlag, München.

Schonlau, M. (2003). Visualizing Categorical Data Arising in the Health Sciences Using Hammock Plots. *In Proceedings of the Section on Statistical Graphics, American Statistical Association*.

Slonim, N., E. Aharoni, and K. Crammer (2013). Hartigan's K-Means Versus Lloyd's K-Means - Is It Time for a Change?. *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*: 1677–1684.

Stark, P.B. (2013). SticiGui. URL `https://www.stat.berkeley.edu/~stark/SticiGui/index.htm`. Last accessed on Mar 02, 2017.

Tan, P.-N., M. Steinbach, and V. Kumar (2005). *Introduction to Data Mining*. Pearson, Edinburgh.

Therneau, T., B. Atikinson, and B. Ripley (2015). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-10 URL `https://CRAN.R-project.org/package=rpart`.

Theus, M. (1997), *MANET: Extensions to Interactive Statistical Graphics for Missing Values*, New Techniques and Technologies for Statistics II: Proceedings of the Second Bonn Seminar. IOS Press, 247–260.

Tidwell, J. (2011). *Designing Interfaces*. O'Reilly Media, Sebastopol, Carl., 2nd edition.

Tufte, E.R. (2001). *The visual display of quantitative information*. Graphics Press, Cheshire, Conn, 2nd edition.

Wegman, E.J. and Q. Luo (1997). High dimensional clustering using parallel coordinates and the grand tour. *Computing Science and Statistics*, 28: 352–360.

Wolf, H.P. (2014). *aplpack: Another Plot PACKage: stem.leaf, bagplot, faces, spin3R, plotsummary, plothulls, and some slider functions*. R package version 1.3.0 URL `https://CRAN.R-project.org/package=aplpack`.

Declaration of Authorship

I hereby confirm that I have authored this Master's thesis independently and without use of others than the indicated sources. All passages which are literally or in general matter taken out of publications or other sources are marked as such.

Berlin, July 18, 2017

Jonas Gärtner